On random gossiping in wireless sensor networks

Dem Fachbereich 18
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.)
vorgelegte Dissertation

von
M.Sc. Zhiliang Chen
geboren am 14.11.1985 in Shandong, V.R.China

Referent:	Prof. Dr.-Ing. Anja Klein
Korreferent:	Prof. Dr.-Ing. Matthias Hollick
Tag der Einreichung:	10 September 2018
Tag der mündlichen Prüfung:	08 Juli 2019

D 17
Darmstädter Dissertation

This work has been funded by the LOEWE research initiative of the state Hesse/Germany within the LOEWE Priority Program Cocoon (www.cocoon.tu-darmstadt.de).

Chen, Zhiliang : On random gossiping in wireless sensor networks
Darmstadt, Technische Universität Darmstadt,
Jahr der Veröffentlichung der Dissertation auf TUprints: 2019
URN: urn:nbn:de:tuda-tuprints-89300
Tag der mündlichen Prüfung: 08.07.2019

Veröffentlicht unter CC BY-NC-SA 4.0 International
https://creativecommons.org/licenses/

Kurzfassung

Diese Dissertation untersucht Random-Gossiping in drahtlosen Sensornetzwerken. Sensoren in drahtlosen Sensornetzwerken erzeugen Messdaten und kommunizieren diese so miteinander, dass die gewünschte Aggregation der Messungen aller Sensoren erreichbar ist. Random-Gossiping ist ein dezentrales Kommunikationsparadigma für drahtlose Sensornetzwerke. Wenn Random-Gossiping im Netzwerk angewendet wird, wacht ein Sensor in zufälliger Weise auf und tauscht Nachrichten mit seinen Nachbarsensoren aus. Kritische Probleme bei der Verwendung von Random-Gossiping für die Aggregation sind die nicht messbare Konvergenz, Verzerrung der Aggregation, die Konvergenzgeschwindigkeit, die durch die Anzahl der Kommunikationen im Netzwerk gemessen wird, und die mögliche Unterstützung mehrerer Anwendungen.

In dieser Dissertation wird ein Sensor als die Integration von Sensorik, Übertragung, Berechnung und Speicherung modelliert. Die Ermöglichung der Kommunikation zwischen Sensoren erfordert ein Cross-Layer-Design, um die Anforderungen an Effizienz und geringen Stromverbrauch zu erfüllen. Um das Cross-Layer-Design zu erleichtern, wird das Konzept des Indicating-Headers vorgeschlagen. Der Indicating-Header dient als die gemeinsame Information, die den Aggregationsstatus der Messung eines bestimmten Sensors in der Nachricht eines anderen Sensors enthält. Daher ist eine direkte Metrik der Konvergenz gegeben. Um die Verzerrung der Aggregation zu überwinden, wird die Speicherkapazität jedes Sensors mit Hilfe der Indicating-Header genutzt. Ein Sensor kann die zuvor in Speicher gespeicherten empfangenen Nachrichten verwenden, um die Verzerrung der Aggregation zu reduzieren. Es wird gezeigt, dass eine Reduktion der Verzerrung erzielbar ist, indem eine Teilmenge der Nachbarsensoren eines Sensors ausgewählt wird, um die Kommunikation durchzuführen.

Um die Konvergenzgeschwindigkeit zu verbessern, werden die Indicating-Headers beim Random-Gossiping vor der Übertragung der Nachrichten, die die Aggregationsdaten enthalten, kommuniziert. Die Information im Indicating-Header ermöglicht dem Sensor, über die Notwendigkeit einer Nachrichtenkommunikation zu entscheiden. Wenn er mit mehreren Nachbarsensoren kommuniziert, verwendet der Sensor den Indicating-Header, um nur eine Teilmenge von Nachbarsensoren für die Kommunikation auszuwählen. Eine Verringerung der Anzahl von Kommunikationen wird erreicht, während die Effizienz der Aggregation erhalten bleibt. Eine weitere Methode zur Verbesserung der Konvergenzgeschwindigkeit wird vorgeschlagen, um Sensoren zu koordinieren, die im Random-Gossiping mehrere Hops von dem Sensor entfernt sind. Wenn die Einschränkung der Netzwerktopologie vorgenommen wird, dass der Sensor und sein Nachbarsensor statisch bleiben, kann Random-Gossiping verbessert werden,

indem die Kommunikation der Indicating-Headers reduziert wird. Darüber hinaus können diese Sensoren, wenn sie sich an topologischen Engpasspositionen des Netzwerks befinden, ihre Nachrichtenkommunikation verschieben, bis die Gruppen von Sensoren, die sie "überbrücken", eine lokale Aggregation erreicht haben. Eine derartige Übertragungsverzögerung, die auf diese Sensoren angewendet wird, reduziert weiter die Anzahl von Kommunikationen im Netzwerk.

Wenn mehrere Anwendungen im Netzwerk ausgeführt werden, muss ein Unterschied hinsichtlich der Anzahl der durchzuführenden Kommunikationen zwischen den Sensoren, die an einer bestimmten Anwendung beteiligt sind, und denen, die nicht beteiligt sind, festgestellt werden. Eine Verfeinerung des Random-Gossiping wird vorgeschlagen, indem sechs verschiedene Szenarien in Bezug auf die Beteiligung eines Sensors und seiner Nachbarsensoren in einer Anwendung betrachtet werden. Der Indicating-Header wird verwendet, um den Sensoren zu ermöglichen, zwischen den sechs verschiedenen Szenarien zu unterscheiden. Die Sensoren, die nicht an der Anwendung beteiligt sind, benötigen nach der Verfeinerung weniger Kommunikationen, während mehr Kommunikationen von den Sensoren, die an der Anwendung beteiligt sind, ausgeführt werden. Hierbei wird die Gesamtzahl der Kommunikationen im Netzwerk beibehalten.

Abstract

This thesis studies random gossiping in wireless sensor networks. Sensors in wireless sensor networks generate measurement data and communicate it with each other such that the desired aggregation involving the measurements of all sensors is achievable. Random gossiping is a decentralized communication paradigm for wireless sensor networks. When random gossiping is applied in the network, a sensor wakes up in a random manner and exchanges messages with its neighbor sensors. Critical problems of using random gossiping for the aggregation are the unmeasurable convergence, the bias of the aggregation, the convergence speed measured by the number of communications in the network, and the support of multiple applications, potentially.

In this thesis, a sensor is modeled as the integration of sensing, transmission, computation, and storage. The enabling of the communications among sensors requires a cross-layer design to meet the efficiency and low power consumption requirements. To facilitate the cross-layer design, the concept of indicating-header is proposed. The indicating-header serves as the shared information containing the aggregation status of the measurement of a particular sensor in the message of another sensor. Therefore, a straightforward metric of the convergence is given. To overcome the bias of the aggregation, the storage capacity at each sensor is explored with the help of the indicating-headers. A sensor can use the previously received messages stored in the memory to cancel the bias in the aggregation. An improvement of the bias cancellation is shown to be achievable by selecting a subset of the neighbor sensors of a sensor to perform the communications.

To improve the convergence speed, the indicating-headers are communicated in the random gossiping before the transmission of the messages containing the aggregation data. The information in the indicating-header enables the sensor to decide on the necessity of message communications. When it communicates with multiple neighbor sensors, the sensor uses the indicating-header to select only a subset of neighbor sensors for communications. A reduction in the number of communications is achieved while the efficiency of the aggregation is maintained. A further method to improve the convergence speed is proposed to coordinate sensors that are multiple hops away from the sensor in the random gossiping. When the constraint to the network topology is made that the sensor and its neighbor sensors remain static, the random gossiping can be improved by reducing the indicating-header communications. Moreover, when sensors are at topological bottle-neck positions of the network, these sensors may defer their message communications waiting for the groups of sensors that they are "bridging"

to have aggregation locally achieved. Such transmission deferment applied to these sensors reduces further the number of communications in the network.

When multiple applications are running in the network, a difference in terms of the number of communications to perform shall be made between the sensors that are involved in a specific application and those that are not. A refinement of the random gossiping is proposed by considering six different scenarios with respect to the involvement of a sensor and its neighbor sensors in an application. The indicating-header is used to enable sensors to distinguish between the six different scenarios. The sensors which are not involved in the application require fewer communications after the refinement while more communications are performed by the sensors that are involved in the application. Meanwhile, the total number of communications in the network is maintained.

Contents

Chapter 1
Introduction

1.1 Wireless sensor networks

In recent years, wireless sensor networks as a technology gain extensive focus from industry and academic studies since they behave as the foundation of the future Internet-of-Things (IoT) [AIM10; XHL14], Industry 4.0 [Böh12] and concepts such as smart city [SKP+11; ZBC+14].

Wireless sensor networks are composed of battery-powered sensors. Sensors communicate using the wireless channel to transmit their own sensed data and receive sensed data from other sensors [ASSC02; AMC07; CMH10; FKK10]. The sensed data of a sensor is a measurement value or a detection output of physical events and quantities from its environment. This definition implies a considerable amount of possible types of sensed data ranging from temperature, humidity to video, and audio recording [ASSC02; AMC07]. The communication of the sensed data requires the networking of sensors. A sensor which is networking with other sensors forming a network is described as a "sensor node". On top of the networking of sensors, a wireless sensor network computes a function which is known to all sensors in the network using all the sensed data. This process is described as data aggregation. The data aggregation could be performed by a defined sensor node or by the involvement of all sensor nodes. Based on the given definitions, there are two primary aspects which shall be analyzed for a wireless sensor network:

- How sensors are networking by defining how and with which other sensors a sensor communicates.
- How a sensor network achieves data aggregations efficiently, concerning both the time-to-convergence and the aggregation completeness.

Three metrics are usually considered to support the analysis:

- The network coverage [CW06] tells in general how to deploy a sensor network with a large number of sensors.

- The network lifetime [ASSC02] provides insight into the energy cost of the communications.
- The convergence speed [AYSS09] quantifies the time needed to achieve the data aggregation.

Many early publications studied a wireless sensor network with simple downloading where the data aggregation is completed only at a gateway node. The gateway node listens to the sensed data from each sensor directly and performs the computing [Arn02; CK03; Arn05; GK05].

The simple downloading requires a direct communication from each sensor node to the gateway node, and it is difficult to extend a network since the extension requires an increase of the transmit power of the sensor nodes which will be deployed with a considerable distance to the gateway node. The coverage of the sensor network is therefore limited.

The network lifetime of the sensor network in the simple downloading is a result of the time duration that batteries of sensors can support communications between sensors and the gateway. In simple downloading, sensors with a larger distance to the gateway consume more energy for communications, resulting in a faster drain of their batteries.

The convergence speed of data aggregation in simple downloading depends on the scheduling of the communications. The scheduling is typically done at the gateway and consumes its computation power. Data aggregation requires the measurement data of every sensor in the network. When the number of sensors increases, the time required for data collection is also increasing.

In order to improve the network coverage while maintaining the network lifetime and the convergence speed of data aggregation, the networking of sensors can be done in an ad-hoc manner [ASSC02; GK05]. Similar to the simple downloading, a wireless ad-hoc sensor network consists of sensors which generate measurement data and a gateway which collects and aggregates all the data. In ad-hoc sensor networks, sensors can communicate with each other, and data aggregations can be performed at sensors. A sensor transmits a message to another sensor which is closer to the gateway topologically, then the receiver sensor aggregates data in the received message and transmits a new message to another sensor. To facilitate the gateway to aggregate all data, techniques such as routing and clustering are necessary to build a routing tree which is rooted at the gateway and branched to all sensors in the network [AK04; RV06]. Because of such ad-hoc manner, the coverage of the sensor network can be expanded

dynamically by introducing new sensors to the network. Since most sensors that are far away only communicate with their neighbor sensors, a low transmit power is necessary at sensors despite the increase of the number of sensors. Therefore, the network lifetime is improved. The convergence speed of data aggregation is no longer constrained by scheduling and computation power at the gateway as the data aggregations are performed along with the message communications between sensors. What is more, the amount of data to aggregate at the same time can be reduced in comparison to the simple downloading.

In general, sensors may have different kinds of measurements at the same time, e.g., temperature, humidity, video, and audio [AMC07]. The gateway of a wireless sensor network may interest in collecting and aggregating different kinds of data using different computations. Therefore, different computations with different types of measurement data are required at sensors in the routing tree. This is named as multiple applications running on a wireless sensor network [AA09].

1.2 Random gossiping

The construction and the maintenance of a routing tree in wireless ad-hoc sensor networks introduce a significant amount of overhead and require a high number of communications in wireless sensor networks with a high number of sensors [AK04]. In recent years, another communication technique named random gossiping to support communications and data aggregations in wireless sensor networks attracts tremendous research focus. In comparison to the simple downloading and ad-hoc sensor network, there is not a specified gateway in the network to collect and aggregate data. Instead, every sensor collects and aggregates data, and the result of the data aggregation taking data from all sensors in the network into account is available at all sensors. Random gossiping requires no centralized scheduling and sensors in the network randomly wake up and initiate communications with sensors in their neighbor. No routing tree is constructed in the network for transmitting messages. The concept of connectivity is used in random gossiping to guarantee that every sensor can exchange messages with any other sensor in the network via one or more communications.

As initially introduced, random gossiping is a decentralized algorithm to solve consensus problems which calculates the average value of the measurement data from all sensors in a wireless sensor network [BGPS04],[BGPS05],[BGPS06], [AYSS09]. It is then extended in the signal processing field as a decentralized processing method

[DBS11]. In those works, the principle of the random gossiping is to update the aggregation data at a sensor by the weighted summation of its current aggregation data and aggregation data received from its neighbor sensors. With proper tuning of the weighting factors, the result of the aggregation data at each sensor will asymptotically approach the desired output. For every weighted summation at a sensor, communications are needed between a sensor and its neighbor sensors. Therefore, the number of communications determines the convergence speed of data aggregation of applications that use random gossiping as a decentralized data aggregation method. Other works such as [SBS12] propose an optimization problem to find the best topology for wireless sensor networks so that the convergence speed is optimally achieved.

In general, random gossiping is a decentralized communication paradigm applicable not only to wireless sensor networks. When there is no central node responsible for the scheduling of communications in a network requiring all-to-all communications, random gossiping is a proper candidate of communication strategies for the network. An example of these networks is mobile networks where mobile phones can communicate with each other via short-range communications using technologies such as Bluetooth instead of via base stations. A message of any mobile phone can in principle be forwarded to any other mobile phones using random gossiping. In other examples such as car-to-car communications [KLS08] and social networking [LM09][RS11][CSA13][WVMX14], random gossiping can also be used as a method when no center exists for scheduling the communications in the network.

1.3 State-of-the-art

1.3.1 Introduction

This section provides a review of the state-of-the-art works that stand as the fundamentals of the work in this thesis. Three topics are covered in this section. In the first topic, we review the random gossiping and its usage in wireless sensor networks on computing the consensus of sensor data. The random gossiping mentioned in the first topic is the communication paradigm considered throughout this thesis. The second topic discusses divisible functions. The divisible functions help to generalize the random gossiping to compute functions other than consensus. It is also the basis of bias-cancellation introduced in this thesis. The third topic covers many current works on running multiple applications in wireless sensor networks.

1.3.2 Random gossiping for consensus

The survey [HHL88] introduces gossiping as a way to solve the information dissemination problem where every node in the network knows a piece of the information. Hence communications are needed between these nodes to achieve the goal that every node gets the complete information in the network, i.e., *everybody knows everything*. The authors highlighted that the number of communications is more general to study than the total time, which is consumed for achieving the goal of gossiping. Meanwhile, principles based on graph theory can be used to reduce the necessary communications between nodes in the network.

Gossiping is a solution of network communications with robustness, simplicity, and scalability since a sensor in the network requires only the information from its direct neighbor sensors [KSSV00]. The communication range of sensors in the network determines the number of their neighbor sensors. In [KKD01], it is shown that the communication range of every sensor in the network determines the convergence speed of data aggregation.

Due to the absence of scheduling of the communications in networks where random gossiping is used, the connection between two sensors in the network at a particular time is probabilistic [KK02]. The probability of the connection can be determined by one over the numbers of communications are needed until a connection between two nodes is established. The work [KK02] considers a network where each node has a message to share with all other nodes. A connection between two sensors is established randomly to exchange messages. Based on a minimum spanning tree using a gossiping mechanism for message delivery, the time that the given number of messages have been exchanged is presented in this work.

The most successful application for random gossiping is to calculate the consensus in wireless sensor networks. A consensus problem involves a) sensors with measurement data and b) a mean value to be calculated and acquired at all sensors. In works [KDG03][BGPS04], sensors communicate data directly with their neighbor sensors. Every sensor generates new data by using the weighted summation of its own data and the data it received from its neighbor sensors. A weighting factor shall be chosen for the weighted summation at each sensor. It is proven that the consensus can be asymptotically achieved by a correct choice of the weighting factors.

The convergence speed of data aggregation in random gossiping is in general slow due to the lack of centralized scheduling. In the consensus problem, a large number of

communications in the network are needed until the data at every sensor equals the real mean value with a relatively small error [BGPS04]. In [BGPS05], the convergence speed of the consensus using random gossiping is proved to be determined only by the second largest eigenvalue of a doubly stochastic matrix, i.e., a matrix whose entries are the probabilities of establishing a connection between two sensors. The sub-gradient of the doubly stochastic matrix can be used to find the weighting factors to guarantee the asymptotic achievement of the consensus [BGPS06].

To improve the convergence speed of the gossiping for consensus problems, the work in [DSW08] combines random gossiping with location-based geographic routing if the knowledge of sensor locations is available at each sensor a priori. A message containing the measurement data is routed to a randomly chosen sensor in the network. The cost of knowing the locations of every sensor is to update and communicate the location information iteratively among sensors.

The convergence speed of consensus is improved when broadcast communication is used in wireless sensor networks. In [AYSS09] one-way broadcast is performed from a sensor to its neighbor sensors instead of the pairwise data exchange of sensors. A weighted summation is applied at each neighbor sensor. In order to ensure the convergence of the consensus, the weighting factor at each sensor shall be calculated.

With clustering techniques, sensors are grouped to form clusters based on parameters such as the location of sensors. Combining random gossiping and clustering can also improve the convergence speed of data aggregation for consensus problem [GBS12]. Data exchanges between clusters are assisted by the sensors which are overlapped by several clusters.

The optimal convergence speed of data aggregation in consensus problems using random gossiping is achievable when a topology optimization is performed [SBS12]. The transmit power at every sensor is optimized to achieve the optimal topology for consensus when the distances between every two sensors are known a priori. When the distance information is unavailable, sensors can only use the same transmit power.

1.3.3 Divisible functions in wireless sensor networks

Besides the consensus problem, computing other applications using random gossiping is of great interest to studies of wireless sensor networks. To combine the random gossiping with computations other than consensus, a definition is needed for general

functions that can be computed using a divide-and-conquer fashion such as consensus for wireless sensor networks.

The functions that can be calculated in wireless sensor networks using a divide-and-conquer fashion are defined as *divisible functions* which are introduced for ad-hoc wireless sensor networks using routing [GK05] [GK06]. The measurement data of sensors are forwarded and aggregated along the routing tree using the divisible functions. In this thesis, we use the concept of divisible functions as the foundation of our work in random gossiping.

In [MS08], [SH08] and[DBS11], random gossiping is explored to calculate divisible functions which can be approximated using methods of consensus. The design of the random gossiping for these functions is ultimately the design and the update of the weighting parameters at each sensor. For example, the application specified in [DBS11] is the resource allocation in cognitive radio networks. A set of sensors exchange the observed channel condition using random gossiping and react to the change of the channel. The weighting factor at each sensor is determined and updated iteratively in order to guarantee the convergence of the data of every sensor.

1.3.4 Multiple Applications in Wireless Sensor Networks

Recent works reveal that multiple applications can be running in one wireless sensor network. A fundamental problem is the sharing of sensors in the network with different applications. The recent works focus on scheduling or slicing as two major solutions.

In scheduling, the use of one sensor by one application is scheduled according to two criteria.

- One criterion considered for scheduling is the resource usage of running one application of a sensor. The resource can be the energy consumption of running a given application [BS03] or the number of sensors required of one application considering the location of the sensors [KMN11; KMN12]. The scheduling is performed by an iterative injection of applications into the network. When injecting one application, the requested additional resource, such as energy consumption or the number of sensors, is estimated. This iterative procedure is carried out until all the intended applications are injected.

- Another criterion for scheduling is to auction based on the quality-of-service (QoS) requirement of an application [EXR+11]. An application is scheduled on one sensor only if the QoS can be fulfilled when it is running on the sensor. According to QoS requirements, applications can be categorized as real-time applications, delay-tolerant applications, or loss-tolerant applications [FKS11] [GHSW11].

When using slicing, sensors in the network are divided into several subsets, where each set supports only one application. Two main criteria are applied to divide the sensors.

- In a Quality of Monitoring criterion, sensors whose measurements of the same physical phenomenon are highly correlated are divided into different subsets [BSLR10; XSC+10]. An example is the temperature measuring of an area by sensors. The measurements of two sensors which are geographically close to each other have a higher correlation in comparison to the measurements of two sensors which are far from each other.

- By computing the geographical area that a set of sensors covers, the subset division of sensors can result in a balanced subset of sensors, i.e., each subset of sensors has a similar coverage area. When applications are running on different subsets of sensors, the coverage area of each application can also be balanced [MZ10; SEH11; SEH12].

When there is a gateway in the wireless sensor network to collect and aggregate data of different applications, a subset which is running one application may also forward the data of another application. It is because a subset of sensors may not be able to connect to the gateway without forwarding the data through another subset of sensors after the slicing is done in the network [JHI07]. When a subset of sensors transmit the data of another subset of sensors towards the gateway, data can be concatenated and forwarded to the gateway [AA09].

If the slicing of the network is unbalanced, an optimal way to assign a subset of sensors to an application can be done based on the requirement of an application on the coverage [RRJ10].

In addition to these topics of multiple applications in a wireless sensor network as mentioned above, sensor buffer management and security are also focused topics. When multiple applications are running simultaneously on a wireless sensor network, the

limited buffer at each sensor could be problematic. To reduce the buffer usage and avoid the buffer overflow, techniques such as code dissemination can be used where the common data such as the measurement or even the program code data could be shared by different applications [LDZ+08]. Concerning the security problem of multiple applications in a wireless sensor network, authors of [LCS13] propose to apply the encryption to the aggregation data of the applications. With the encryption, an application cannot access the content of other applications running on the same sensor.

1.4 Open Issue

In this section, open issues concerning applying random gossiping in wireless sensor networks are listed with the foundation of the works reviewed in Section 1.3.

In Section 1.3, it shows that random gossiping has been mainly applied to solve consensus problems in wireless sensor networks. In such a category of problems, the design of random gossiping is to design the weighting factors at each sensor. However, using random gossiping for computing an arbitrary divisible function has not been discussed. Therefore, the following questions arise:

- How to apply random gossiping to wireless sensor networks to compute arbitrary divisible functions which cannot be modeled by weighted summation?
- How to guarantee the convergence at every sensor such that the data aggregation yields the desired results for general applications which do no use the weighted summation?

Since random gossiping requires no scheduling of communications of sensors, it, in general, requires a large number of communications until the convergence of the data aggregation is achieved. Many works on random gossiping that we have reviewed in Section 1.3 addressed this problem. However, all these works focused on improving the convergence speed of data aggregation of the consensus problem. Therefore, the solution is not general. Moreover, there is no significant improvement in the performance of those works. A problem then arises:

- How to improve the convergence speed of data aggregation for random gossiping in general?

In some wireless sensor networks, assumptions on the topology are made such as static sensor locations. These assumptions lead to the following problem:

- How to make use of the assumptions on the topology to improve the performance of random gossiping?

So far, using random gossiping in a wireless sensor network with multiple applications is not a well-discussed topic. An interesting problem can be raised:

- How to apply random gossiping to wireless sensor networks that can support and run multiple applications?

1.5 Contributions of the thesis

This section gives a brief statement of the main contributions which jointly or individually addresses the open issues stated in Section 1.4. The contents are described according to the order of the open issues presented in Section 1.4.

- Chapter 2 presents the network model of the wireless sensor network that is applied throughout this thesis. In order to apply random gossiping to compute divisible functions such that generic applications can be supported, we propose a cross-layer design of wireless sensor networks to address the sharing of the information crossing different layers. More importantly, the concept of Indicating-Headers (I-Headers) is proposed in this chapter. I-Header serves as a message header to record the aggregated data at each sensor. It is also shared information in the cross-layer design.

- The concept of bias reduction in random gossiping is introduced in Chapter 3. Bias reduction is based on the concept of I-Headers introduced in Chapter 2. It uses the capability of sensors to store old messages and helps to achieve the convergence of the data aggregation of using random gossiping for divisible function calculation. In this chapter, two bias cancellation methods are proposed. In the second method, the bias reduction considers as well the selection of a subset of the neighbor sensors from a sensor to perform communications.

- Chapter 4 addresses the reduction of the convergence time of data aggregation using random gossiping in wireless sensor networks. Based on how to communicate with neighbor sensors, the types of humble sensors and greedy sensors are introduced. The algorithms used in random gossiping with I-Header and bias reduction are proposed to reduce the convergence time. Moreover, we discuss the possibility and introduce algorithms to combine random gossiping and routing algorithms that are used in wireless ad-hoc sensor networks, in order to increase the convergence speed of the data aggregation in the network.

- In Chapter 5, the assumption of a static topology of the wireless sensor network is made. An improved random gossiping approach that makes use of a proposed algorithm is given to reduce the convergence time of data aggregation by reducing the communications of I-Headers. We propose a method, called transmission deferment, that enables particular sensors in the network to delay their communications with their neighbors in order to achieve faster convergence of data aggregation with fewer communications in the whole network.

- In Chapter 6, we consider wireless sensor networks that may support multiple applications. Refined algorithms for random gossiping are proposed taking into account that not all sensors are involved in an application. The proposed random gossiping algorithm considers different scenarios based on whether and how many neighbor sensors are involved in an application when a sensor communicates with them. By doing this, the number of communications performed by the sensors which are not involved in an application is reduced.

The conclusions of this thesis are in Chapter 7. A short outlook is provided for possible future extensions of the works in this thesis.

Chapter 2

Modeling of wireless sensor networks and random gossiping

2.1 Introduction

In this chapter, this first topic covered is the model of sensors by introducing the components of sensors and the network model of the wireless sensor networks where we address topics of the deployment of sensors, the communication range, the neighbor sensors and the connectivity of wireless sensor networks. The second topic discusses random gossiping and divisible functions. The random gossiping will be generalized such that it can be applied to applications with divisible functions in wireless sensor networks. Thirdly, a cross-layer design of wireless sensor networks is presented. In the cross-layer design, we discuss what information should be shared by different layers. As a new contribution in this thesis, the concept of indicating headers in wireless sensor networks is introduced. The indicating headers are used to share information across different layers in the cross-layer design.

This chapter is organized as follows. In Section 2.2, the sensor model and the network model are presented. Section 2.3 discusses random gossiping, divisible functions, and the generalization of random gossiping to support the computation of divisible functions in the wireless sensor networks. Section 2.4 introduces the cross-layer design of wireless sensor networks and the concept of indicating headers which is used throughout the remainder of this thesis. Parts of the contents of this chapter have been published by the author of this thesis in [CKK13b].

2.2 Modeling of sensors and wireless sensor networks

2.2.1 Sensor and its components

Throughout this thesis, we use v to indicate a sensor in a wireless sensor network. Each sensor in the wireless sensor network is assumed to consist of four components according to their functionality, as shown in Figure 2.1.

- Sensing component generates the measurement of one or more physical phenomena by sensing the environment. The sensing component transforms the measurement into *measurement data*, which is represented by a finite sequence of bits and outputs it to the computing component. The measurement data that is transformed from the measurement of a specific sensor v is called *the measurement data of sensor* v.

- Transceiver component is responsible for transmitting frames of bits generated from the computing component to other sensors or receiving frames of bits from other sensors and forward to the computing component. In the remainder of this thesis, a frame of bits that are transmitted and received between sensors is named as *a message*.

- The memory component is to provide the capability of storage at the sensor. It stores various kinds of data generated by the computing components, e.g., the previously transmitted or received messages.

- The computing component is the central component of a sensor since it connects the sensing, the memory, and the transceiver components. It configures the sensing component to determine what phenomenon to measure, gets measurement data from the sensing component, creates messages for transceiver components and generates different kinds of data that could be stored in the memory component. In this thesis, *aggregation data* denotes the output of computations, which involves measurement data of at least one sensor. The message of a sensor is generated by the computing component, which encapsulates aggregation data and other information such as message header which will be discussed later in this chapter.

The arrows that connect components show internally in a sensor the direction of the communications between each of them. The sensing component gets the configuration from the computing component and sends the measurement data to it. The computing component can send and receive messages from the transceiver component as well as store and load data from the memory component.

2.2.2 Wireless sensor networks

Individual sensors can communicate wirelessly with each other using their transceiver components. In this thesis, we define a wireless sensor network by a graph expressed using a tuple $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In the tuple, $\mathcal{V}$ is the set of sensors. If the wireless sensor

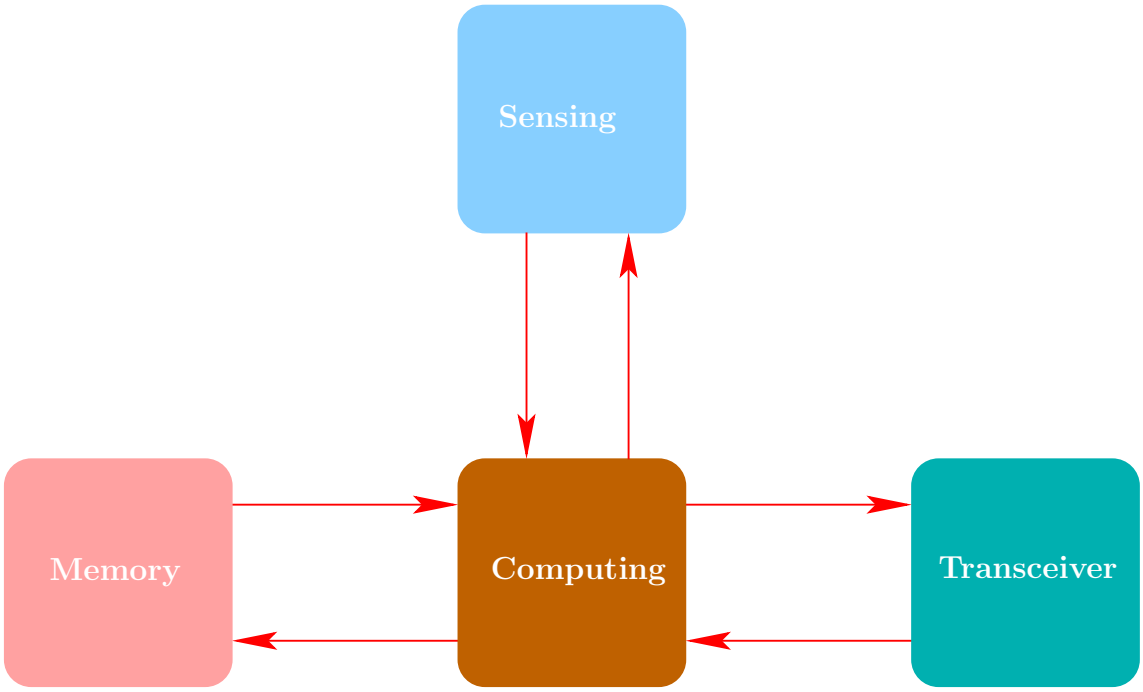

Figure 2.1. Components of a sensor

network consists of N sensors, these sensors can be distinguished in the set $\mathcal{V}$ as $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$. We denote a sensor in network $\mathcal{G}$ by using $v_i \in \mathcal{V}$ where i is an integer number taking values from $\{1, 2, \cdots, N\}$. Each sensor $v_i \in \mathcal{V}$ is assumed to be programmed with a unique ID, and no two sensors are sharing the same ID. In the tuple, $\mathcal{E}$ is the set of all connections in the network.

Throughout this thesis, we assume that the wireless sensor network is in a finite squared area geographically. Without loss of generality, let the bottom left corner of the finite squared area be the origin of the Cartesian coordinate system. The location of a sensor v_i in the wireless sensor network is expressed by a tuple (x, y), where x is the location on the x-axis and y is the location on the y-axis in the coordinate, as shown in Fig. 2.2.

Let a constant D denote the limit of the finite squared area that is limited in both x-axis and y-axis. All N sensors in the wireless sensor network are uniformly randomly deployed with the probability density function

$$p(x) = \begin{cases} \frac{1}{D} & \text{for } x \in [0, D] \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

and

$$p(y) = \begin{cases} \frac{1}{D} & \text{for } y \in [0, D] \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

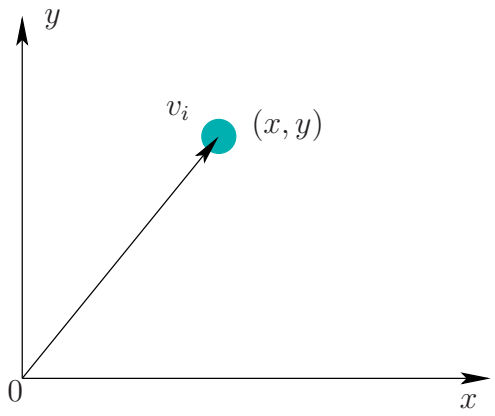

Figure 2.2. Deployment of sensor v_i

We assume that the wireless channels between sensors have only pathloss attenuation, and every sensor $v_i \in \mathcal{V}$ uses the same transmission power. Therefore, a sensor can communicate with another sensor if their distance defined in the Cartesian coordinate system is smaller than a threshold variable d, where d is defined as the *communication range* of sensors in the wireless sensor network. This assumption employs the SNR-model that a sensor can correctly decode the information from another sensor only when the Signal-to-Noise Ratio (SNR) of the received signal is larger than a given threshold. We define one *successful message communication* such that a sensor transmits a message and the targeted sensor or sensors successfully receive and decode the message. Base on the assumption above, the successful communications between any two sensors are only determined by the distance between them. If the distance d_{ij} between sensor $v_i \in \mathcal{V}$ and sensor $v_j \in \mathcal{V}$ is smaller than the communication range d, we say a connection between v_i and v_j, denoted by e_{ij}, exists. The set $\mathcal{E}$ of connections in the graph $\mathcal{G}$ is therewith formally defined as $\mathcal{E} = \{e_{ij} | d_{ij} < d, v_i \in \mathcal{V}, v_j \in \mathcal{V}\}$.

Neighbor sensors of a sensor v_i are sensors whose distance to sensor v_i is less than d. We denote the set of neighbor sensors of sensor v_i by the set $\mathcal{N}_i$ whose cardinality is N_i. For every sensor $v_j \in \mathcal{N}_i$, the connection e_{ij} exists in $\mathcal{E}$. A sensor $v_j \in \mathcal{N}_i$ and the sensor v_i are neighbor sensors to each other. In Figure 2.3 as an example, sensor v_i is in the center of the dashed circle with a radius d. All sensors v_{j_1}, v_{j_2} and v_{j_3} enclosed with the dashed circle except v_i are the neighbor sensors of sensor v_i. Sensors v_{k_1}, v_{k_2} and v_{k_3} that are outside the dashed circle are not neighbor sensors of v_i. There are two communication types from sensor v_i to its neighbor sensors $\mathcal{N}_i$, broadcast, and unicast. In broadcast, sensor v_i can transmit a message to all sensors in $\mathcal{N}_i$ using one successful transmission. In contrast, in unicast, sensor v_i communicates with only one sensor in $\mathcal{N}_i$ in one successful transmission. Therefore, N_i successful transmissions are needed in unicast such that v_i can transmit its message to all sensors in $\mathcal{N}_i$.

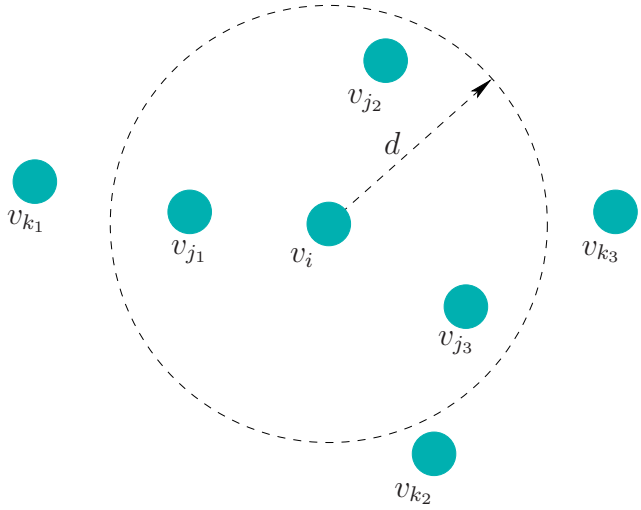

Figure 2.3. Sensor v_i and its neighbor sensors

A preparatory condition for our work in wireless sensor networks is to guarantee the connectivity. The connectivity can be intuitively understood as the existence of a path consisting of a set of intermediate sensors between any two sensors, where the two sensors are not neighbor sensors to each other. However, the connectivity cannot be guaranteed solely by making sure that every sensor $v_i \in \mathcal{V}$ has a non-empty $\mathcal{N}_i$.

A method to check the connectivity is to use the idea from the spectral graph theory [Chu97]. Let $\boldsymbol{A}$ be a square matrix whose entry a_{ij} on the i-th row and j-th column is 1 if and only if the connection between sensor v_i and v_j in the network $\mathcal{G}$ exists, i.e., $e_{ij} \in \mathcal{E}$. The diagonal elements in $\boldsymbol{A}$ are all zeros since we assume that a sensor shall not be its own neighbor. Let another matrix $\boldsymbol{D}$ be a diagonal matrix whose i-th diagonal entry equals N_i, the number of neighbor sensors of sensor v_i. The Laplacian matrix of the network $\boldsymbol{L}$ is defined by $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$. According to the spectral graph theory and its corresponding applications, the connectivity of the network is guaranteed as long as the second smallest eigenvalue of $\boldsymbol{L}$, denoted by λ_2, is greater than 0 [Chu97; SBS12]. Throughout this thesis, we always assume a connected wireless sensor network where the communication range d is large enough to guarantee $\lambda_2 > 0$.

2.3 Random gossiping

2.3.1 Introduction

Random gossiping is a decentralized communication paradigm for wireless sensor networks. In comparison to routing and clustering, random gossiping does not construct

hierarchical infrastructures such as a routing tree or sensor clusters in wireless sensor networks [BGPS06]. Communications in random gossiping are only local communications between sensors and their neighbor sensors. Therefore, random gossiping is robust against link failure. If routing is applied in the network, link failure in the routing tree will cause the failure of the data aggregation [AYSS09].

A slotted time structure is usually assumed in random gossiping [BGPS06]. In the slotted time structure, a sensor wakes up when its randomly initiated timer times out. If the timers at all sensors are asynchronous, the possibility of two sensors initiating communications at the same time is zero. Therefore, we assume that only one sensor in the wireless sensor network wakes up at a time and initiates communications with its neighbor sensors. With this assumption, the interference is not considered in the wireless sensor network using random gossiping.

2.3.2 Random gossiping for consensus

The most successful application of random gossiping is to calculate the consensus of the measurement data of all sensors. Initially, all sensors $v_i \in \mathcal{V}$ have their aggregation data x_i equal to their measurement data, i.e., $x_i = s_i$. By applying random gossiping, the aggregation data at each sensor in the wireless sensor network converges to the mean value $\bar{s} = \frac{1}{N}\sum_{i=1}^{N} s_i$ of all measurement data asymptotically.

There are two different ways to perform random gossiping to calculate consensus in wireless sensor networks. Firstly, as discussed in [BGPS04], [BGPS05] and [BGPS06], sensor v_i wakes up and initiates communication with only one of its neighbor sensors $v_j \in \mathcal{N}_i$. A weight factor of $\frac{1}{2}$ for the weighted summation is chosen such that after communication, sensor v_i and sensor v_j update their aggregation data as $x_i := \frac{1}{2}(x_i + x_j)$ and $x_j := \frac{1}{2}(x_i + x_j)$. It is proven both in [BGPS06] and [SBS12] that the convergence time, measured in terms of the number of communications is upper and lower bounded by values determined by the second smallest eigenvalue λ_2 of the Laplacian matrix $\boldsymbol{L}$.

The second way of using random gossiping to calculate consensus is the broadcast random gossiping, which takes advantage of the broadcast nature of the wireless communications [AYSS09]. When a sensor v_i initiates the communications, it sends its aggregation data x_i to all its neighbor sensors $v_j \in \mathcal{N}_i$. All the neighbor sensors $v_j \in \mathcal{N}_i$ update their aggregation data by weighted summation with their own aggregation data and the received aggregation data with the weighting factor being $\frac{1}{2}$, i.e., $x_j = \frac{1}{2}(x_i + x_j)$. The bound of the convergence time, which is measured in the number

of communications until convergence as proven in [AYSS09] is determined as a function of the number of sensors N.

2.3.3 Divisible functions

In order to extend the range of applications that random gossiping can support, we use the concept of the divisible functions introduced in [GK05]. Intuitively, a divisible function specifies a type of function whose parameters can be aggregated gradually. Such gradual aggregation enables a *divide-and-conquer* fashion of calculating the function in wireless sensor networks [GK05].

An application in the wireless sensor network with N sensors can be defined by using a set $\mathcal{F}$ of divisible functions. The set $\mathcal{F}$ contains N functions $f_l \in \mathcal{F}$, $l = 1, 2, \cdots, N$ where the subscript l means the function f_l takes l input parameters. Let $\mathcal{S} = \{s_1, s_2, \cdots, s_N\}$ denote the set of measurement data of all sensors. If a partition of $\mathcal{S}$ divides the total N measurement data measured by the N sensors, respectively, into L mutually exclusive sets, this partition can be denoted by $\Pi(\mathcal{S}) = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_L\}$, where $\mathcal{S}_l$ is the l-th set. The union of all sets fulfills

$$\cup_{i=1}^{L} \mathcal{S}_i = \mathcal{S}. \tag{2.3}$$

Let a vector $\boldsymbol{s}$ collect all measurement data in $\mathcal{S}$ and let vector $\boldsymbol{s}_{\mathcal{S}_k}$ denote all measurement data in set $\mathcal{S}_k$, the cardinality of set $\mathcal{S}_k$ is denoted by l_k. The measurement data in the vectors $\boldsymbol{s}$ and $\boldsymbol{s}_{\mathcal{S}_k}$ are ordered according to the increment of the index of the sensor where the measurement data is generated. In order to calculate a divisible function $f_N(\boldsymbol{s})$, the function whose parameters are the data from each set $f_{l_k}(\boldsymbol{s}_{\mathcal{S}_k})$ is calculated at first, then a combination with an auxiliary function $g^{\Pi(\mathcal{S})}$,

$$f_N(\boldsymbol{s}) = g^{\Pi(\mathcal{S})}\left(f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}), f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}), \cdots, f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) \tag{2.4}$$

is performed to get $f_N(\boldsymbol{s})$ [GK05].

In (2.4), $f_{l_k}(\boldsymbol{s}_{\mathcal{S}_k})$ can be calculated by further partitioning $\mathcal{S}_k$. The partition can be done until there is only one measurement data in each group.

Divisible functions cover most of the possible functions that applications in wireless sensor networks may utilize, such as downloading, histogram, sum, average, mode, max, min. In the following, we list the auxiliary functions for these commonly used divisible functions that are mentioned in [GK05].

- For downloading function, where $f_N(\boldsymbol{s}) = \boldsymbol{s}$, the auxiliary function is

$$g^{\Pi(\mathcal{S})}\left(f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}), f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}), \cdots, f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) = [\boldsymbol{s}_{\mathcal{S}_1}, \boldsymbol{s}_{\mathcal{S}_2}, \cdots, \boldsymbol{s}_{\mathcal{S}_L}]\,. \tag{2.5}$$

- The histogram function calculates the occurrence of measurement data falling into a certain range, so-called *a bin*. Let $(\tau_i^l, \tau_i^u]$ denote the range of the i-th bin, i.e., $f_N(\boldsymbol{s}) = [\tau_1(\boldsymbol{s}), \tau_2(\boldsymbol{s}), \cdots, \tau_\chi(\boldsymbol{s})]$, where χ is the total number of bins for the histogram calculation and $\tau_i(\boldsymbol{s}) = |\{j : s_j \in (\tau_i^l, \tau_i^u]\}|$ returns the number of parameters in $\boldsymbol{s}$ that falls into the i-th bin. For the histogram function, the auxiliary function is

$$g^{\Pi(\mathcal{S})}\left(f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}), f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}), \cdots, f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) = f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}) + f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}) + \cdots + f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\,. \tag{2.6}$$

- The sum function $f_N(\boldsymbol{s}) = \sum_{i=1}^N s_i$ has the auxiliary function

$$g^{\Pi(\mathcal{S})}\left(f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}), f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}), \cdots, f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) = f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}) + f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}) + \cdots + f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\,. \tag{2.7}$$

- The auxiliary function of the average function $f_N(\boldsymbol{s}) = \frac{1}{N}\sum_{i=1}^N s_i$ is

$$\begin{aligned} & g^{\Pi(\mathcal{S})}\left(f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}), f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}), \cdots, f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) \\ = \; & \frac{1}{l_1 + l_2 + \cdots + l_L}\left(l_1 f_{l_1}(\boldsymbol{s}_{\mathcal{S}_1}) + l_2 f_{l_2}(\boldsymbol{s}_{\mathcal{S}_2}) + \cdots + l_L f_{l_L}(\boldsymbol{s}_{\mathcal{S}_L})\right) \end{aligned} \tag{2.8}$$

- The mode function that gives the value occurs most frequently applies the histogram function to compute the output. The same auxiliary function as the histogram function will be used to the mode function.

- The max function $f_N(\boldsymbol{s}) = \max_i s_i$ has an auxiliary function being identical to the max function itself.

- The min function $f_N(\boldsymbol{s}) = \min_i s_i$ has an auxiliary function being identical to the min function itself.

2.3.4 Random gossiping for divisible functions calculations

When random gossiping is used to calculate an arbitrary divisible function, the weighed summation cannot be used because the divisible function is not averaging the measurement data, generally. The computations at each sensor follow (2.4). Therefore, instead of approaching the desired results asymptotically as in consensus, the computation of divisible functions finishes when all measurement data are taken into the computation, as shown in Fig. 2.4.

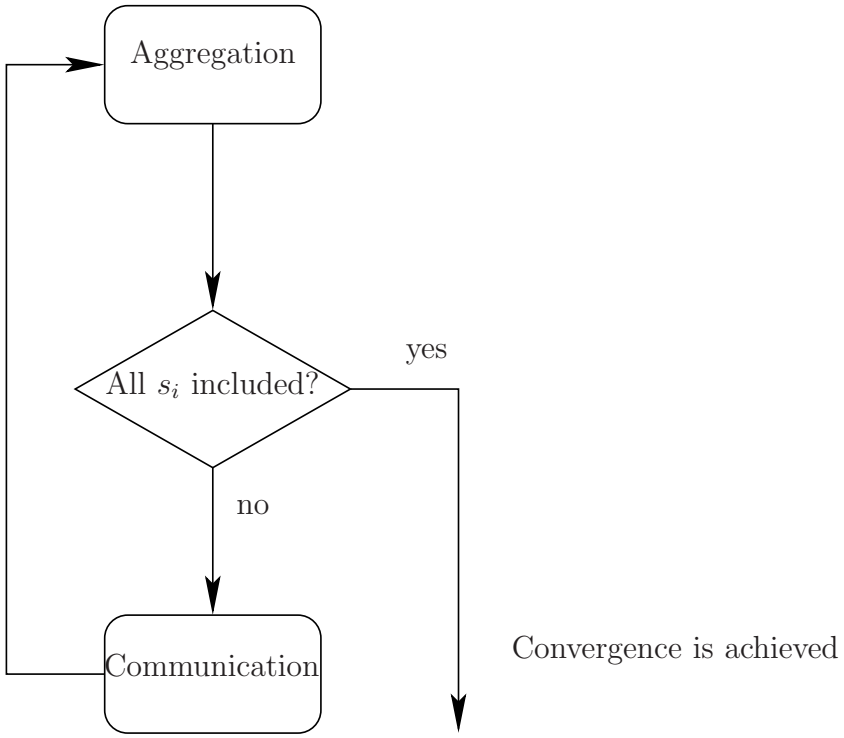

Figure 2.4. Termination of random gossiping to calculate divisible functions

If communications between sensors bring new measurement data which has not been aggregated at a sensor, the aggregation is performed. When the aggregation has already involved all measurement data at every sensor, the convergence of random gossiping is achieved.

2.4 Cross-layer design and indicating headers

2.4.1 Cross-layer model

In order to support generic applications in wireless sensor networks where random gossiping is applied, the communications of sensors should be independent on which application is running in the wireless sensor network. However, information needs to be shared between the application layer where divisible functions are computed to aggregation and the network layer where sensors communicate their messages with each other.

Figure 2.5 shows the cross-layer model considered in this thesis. The application layer provides aggregation data to the network layer to construct messages for communi-

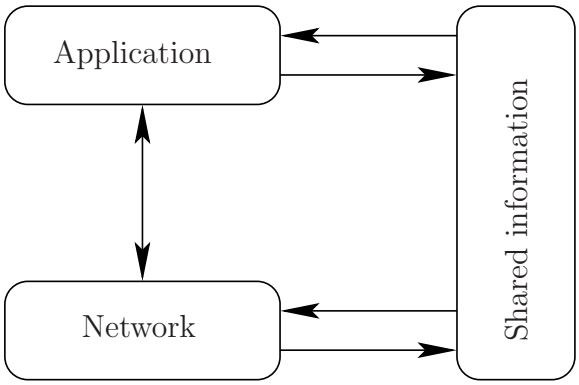

Figure 2.5. Cross-layer model

cations between sensors. Meanwhile, the data aggregations in the application layer modify the information that is shared with the network layer. The network layer determines how communications should be carried out using the shared information. When messages are received from other sensors, the network layer will update the shared information accordingly and provide aggregation data encapsulated in the message to the application layer. The application layer takes the shared information to perform appropriate data aggregation.

Based on the discussions above, the following criteria shall be applied to the shared information:

- the shared information should be involved when the data aggregation is applied in the application layer,
- the shared information should be used by the network layer to decide what information to communicate between the sensor and its neighbor sensors, and
- the shared information should be generic for different kinds of applications.

2.4.2 Indicating headers

In this subsection, we introduce the concept of *Indicating-Headers*. Indicating-Headers (I-Headers) serve as the cross-layer information between the application layer and the network layer. We use I-Headers as the control information in wireless sensor networks where random gossiping is applied. Figure. 2.6 depicts the cross-layer model with I-Header.

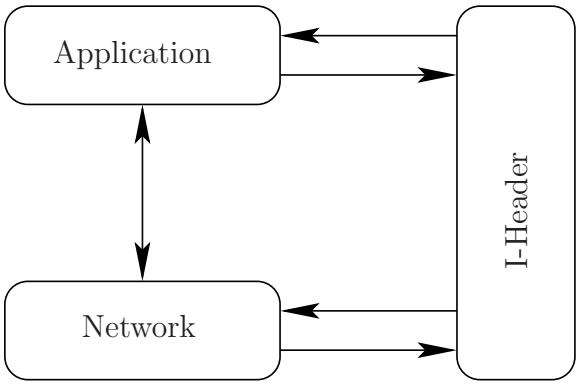

Figure 2.6. Cross-layer model with I-Header

An I-Header is a fixed-length bit sequence paired with every message that is communicated between sensors. For a wireless sensor network with N sensors, the I-Header of a message shall have N bits. The I-Header of the message currently computed at sensor v_i is denoted by $\boldsymbol{I}_i$. If sensor v_i has aggregated the measurement data generated by the sensing component of sensor $v_j, j = 1, 2, \cdots, N$, the j-th bit in $\boldsymbol{I}_i$, denoted by $\boldsymbol{I}_i(j)$ is 1, otherwise 0. Throughout this thesis, we assume that the necessary encodings for error protection are applied for the communications in the network. However, this aspect will not be considered in the discussions due to the reason that the encoding does not provide additional information to which are already provided in the message and the I-Header at each sensor.

Based on this definition, the I-Header tells only whether the measurement data of a sensor has been aggregated in the aggregation data encapsulated in the message without showing the duplication. In random gossiping, this may lead to a so-called *bias problem*, which will be addressed in the next chapter. Additionally, the I-Header of the message currently computed at sensor v_i will be changed as soon as new measurement data is aggregated in the message.

We define a function Θ taking an I-Header as the parameter, and the output is a set collecting the IDs of the sensors defined in Section 2.2 at which the measurement data are generated. For example, if the I-Header of the current message at sensor v_i is $\boldsymbol{I}_i = [1, 0, 0, 1]$, the function output shall be $\Theta(\boldsymbol{I}_i) = \{1, 4\}$. It expresses that the measurement data contained/aggregated in the current message is generated by the sensors whose numerical IDs are 1 and 4 in the wireless sensor network under consideration.

Generically, we denote the set $\Theta(\boldsymbol{I}_i)$ by $\mathcal{S}_i^{\mathrm{i}}$, i.e., $\mathcal{S}_i^{\mathrm{i}} = \Theta(\boldsymbol{I}_i)$, where the superscript i indicates that it is an index set in contrast to set $\mathcal{S}_i$ which collects all the data that

sensor v_i has aggregated in its message. It is straightforward to see that function Θ has an inverse function Θ^{-1}, which takes a set of indices as the parameter and outputs an I-Header, i.e., $\boldsymbol{I}_i = \Theta^{-1}(\mathcal{S}_i^{\mathrm{i}})$.

In this thesis, for a wireless sensor network with N sensors in total, the following assumptions are made:

- Assumption 1: Each sensor has a unique numerical ID such that it can be distinguished from other sensors in the wireless sensor network.
- Assumption 2: There is a function that can map the sensor ID uniquely to a bit position in the N-bit I-Header.
- Assumption 3: The mapping function is known by all sensors.

Under these assumptions, when sensor v_i has not yet aggregated the measurement data from other sensors in the network, applying the mapping function to the message of sensor v_i will result in its own ID i.

Per definition, the I-Header and the corresponding function Θ consider only sensor networks with a fixed number of sensors. In practical applications of wireless sensor networks, there are two possible cases which may invalidate this consideration, the leaving of sensors and the joining of new sensors.

If a sensor disconnects from the wireless sensor network and the connectivity of the network is still maintained, the following two sub-cases can be considered separately.

- If the disconnected sensor has already communicated with its neighbor sensors, its measurement data will be *preserved* as it has been aggregated in the aggregation data in the message of other sensors. In the I-Header of the neighbor sensors which aggregated the measurement data of the disconnected sensor, the corresponding bit of the disconnected sensor will be 1.
- If the disconnected sensor has not yet communicated with its neighbor sensors, its data will be lost permanently. In this case, none of the sensors in the wireless sensor network will aggregate the measurement data of the disconnected sensor. According to the definition of the random gossiping, this situation may lead to permanent communications in the network. In order to prevent this, the sensor network shall provide a measure to stop the communication when the I-Headers of the messages of all sensors in the network remain unchanged for some time.

There are two strategies to handle the situation in the case of new sensors joining the wireless sensor network.

- The first strategy assumes that the length of I-Headers designated to the wireless sensor network should be larger than the number of sensors in the network. When mapping the ID of a sensor in the network to a bit position in the I-Header, there will be given bits in the I-Header being 0. These bit positions can be used for new joining sensors as long as the total number of sensors after new sensors joining the network is smaller than the length of the I-Headers.
- The second strategy ties the newly joined sensor to one of its neighbor sensors, e.g., sensor v_i. The joined sensor will communicate only with sensor v_i. Sensor v_i becomes a delegate of the joined sensor to communicate with other sensors. In this case, there is no extension in the I-Header requested. However, extra information might be needed to indicate that the message of sensor v_i contains the aggregation data that aggregates the measurement data of two sensors. This solution will not work if sensor v_i has already communicated with other sensors, i.e., data s_i has already been aggregated in the messages of other sensors. Since v_i is a delegate of the newly joining sensor, the aggregation data contained in the message of sensor v_i also aggregates the measurement data of the joining sensor. When the other sensors in the network have already aggregated the measurement data s_i, there will be two "versions" of s_i in the network after new sensors joined the network.

The two methods above provide potential solutions to handle the disconnection of sensors or the joining of new sensors in the network.

Two assumptions are made in the remainder of this thesis:

- Assumption 4: There are no joining and leaving sensors in the wireless sensor network.
- Assumption 5: The total number of sensors denoted by N is the maximum number of sensors the ID sequence, as well as the function Θ can support.

With I-Headers, the communications in wireless sensor networks can be categorized into the communications of application messages and the communications of the I-Headers. Control information in I-Header is exchanged in the network and provides

the information for the network to control the behavior of each sensor. Additionally, meaningless communications of application messages are reduced. This reduction is beneficial when the size of the messages exceeds the size of the I-Header significantly. It becomes useful in realistic cases when sensors are designed to sense diverse kinds of information ranging from temperature and humidity to video or audio clips.

Throughout this thesis, an additional assumption is made:

- Assumption 6: The size of the application messages is significantly larger than the length of the I-Header.

2.5 Summary

In this chapter, the model of sensors and the model of wireless sensor networks have been presented. We discussed random gossiping and its extension to calculate divisible functions. More importantly, the cross-layer design and the concept of I-Headers have been introduced which are the foundations of our work in the coming chapters. Last but not least, six assumptions have been made which will be used in the remainder of this thesis.

Chapter 3
Bias reduction

3.1 Introduction

As discussed in Chapter 2, random gossiping requires a large number of communications typically in order to achieve the convergence of data aggregation for computing a function that takes the data from all sensors as parameters. Many works such as [BGPS06] and [SBS12] propose methods to increase the convergence speed of data aggregation by tuning the topology of the wireless sensor networks. These methods have two problems:

- The algorithms are centralized off-line methods. Centralized solutions compromise the robustness and the flexibility of random gossiping in wireless sensor networks because the sensors are required to be deployed at the exact topological positions that the algorithms assume. The off-line solutions imply that the status of sensors in the wireless sensor network should be uploaded to a central unit, and the optimized topology shall then be downloaded to all sensors after the optimization is completed at the central unit. These are impractical solutions for real-world wireless sensor networks.
- Each algorithm proposed in works such as [BGPS06] and [SBS12] provides optimization to increase the convergence speed of data aggregation of only one type of application, e.g., the consensus. The optimization algorithms take the mathematical expression of the computation functions used in the data aggregation into account, e.g., sum-and-divide in consensus. The resulting topology is therefore only optimized to support a narrow spectrum of applications.

As a foundation to solve these problems, a cross-layer design is proposed in Chapter 2 for wireless sensor networks where random gossiping is applied. Specifically, the concept of I-Header is introduced as the shared information between the application layer and the network layer. The application layer uses the I-Header to perform the appropriate data aggregation, and the network layer uses it to determine how communications should be carried with other sensors. An I-Header is always paired with a message that is communicated between sensors. Other sensors can know the aggregation data contained in the message through the information given in the I-Header.

The cross-layer design and I-Header are used to support the random gossiping for applications where the computation functions are divisible functions. As mentioned in Chapter 2, the I-Header tells only whether the measurement data of a sensor has been aggregated in the aggregation data without showing the duplications. In random gossiping, this can lead to *bias*. In this chapter, the concept of bias in random gossiping is introduced. Algorithms for reducing and eliminating the bias are proposed for the bias cancellation.

The remainder of this chapter is organized as follows. In Section 3.2, the definition of the bias in random gossiping is given. Section 3.3 introduces a bias reduction algorithm. An improved bias-reduction algorithm with joint sensor selections is discussed in Section 3.4. Section 3.5 summaries this chapter. Parts of the content of this chapter have been published by the author of this thesis in [CKK13b] and [CKK14a].

3.2 Bias in random gossiping

3.2.1 Definition of bias

In this section, the definition of the bias in random gossiping is introduced. A sensor aggregates measurement data by receiving the messages from other sensors.

However, during the aggregation, the measurement data from a sensor may be aggregated more than once. In this thesis, the bias can be formally defined as follows.

- The bias of the aggregation data at a sensor v_i is defined as the total number of duplicate measurement data $s_j, j = 1, 2, ..., N$ in the aggregation data.

For example, the aggregation data at sensor v_i has aggregated measurement data s_i, s_j, and s_k. The measurement data s_i has been aggregated twice, the measurement data s_j has been aggregated three times, and s_k has only been aggregated once. The bias of the aggregation data at sensor v_i is $1 + 2 + 0$.

Furthermore, we define the bias of a measurement data s_j in the aggregation data at sensor v_i as the duplication of measurement data s_j that has been aggregated at sensor v_i. In the example, in the aggregation data of sensor v_i the bias of s_i is 1, the bias of s_j is 2, and the bias of s_k is 0.

In random gossiping, the bias of the aggregation at a sensor may vary continuously during the communications of the messages containing aggregation data of other sensors.

In this thesis, the following two situations which result in bias in the aggregation data are considered. Figure 3.1 shows an example of the first situation. In the example, the divisible function is $f_{l_k}(\boldsymbol{s}_{\mathcal{S}_k}) = \boldsymbol{s}_{\mathcal{S}_k}$, and the arrows show the spreading of the measurement data of sensor v_i. The aggregation data of a sensor v_i is spread to its neighbor sensors v_j and v_k. As the aggregation and spreading continue, another sensor v_s in the wireless sensor network may receive messages from its neighbor sensors v_l and v_m where s_i is aggregated at both sensors. As a result, a bias of the measurement data s_i exists in the aggregation data at sensor v_s.

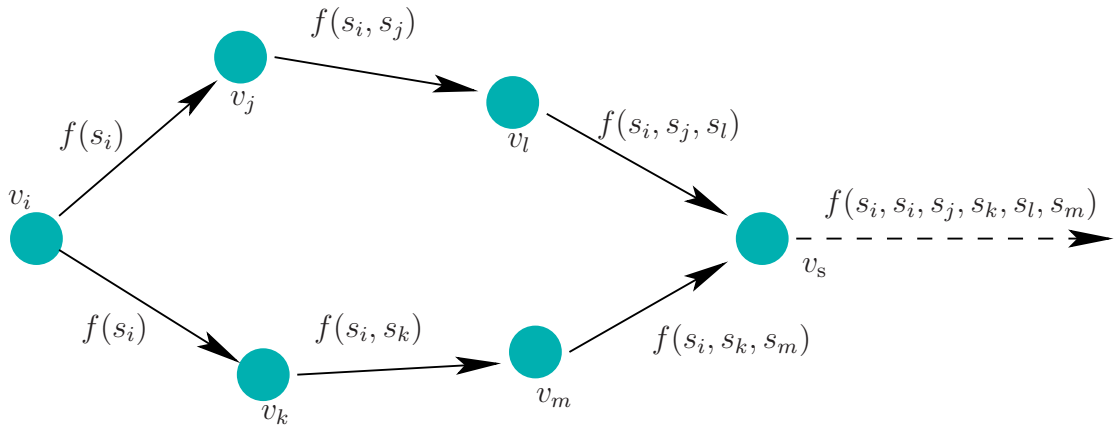

Figure 3.1. Duplication by aggregation from different neighbor sensors

In the second situation, a sensor may receive aggregation data where its own measurement data has been aggregated.

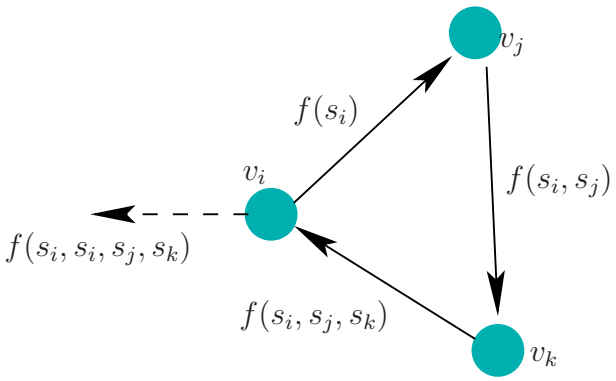

Figure 3.2. Duplicated data in gossiping protocol

An example is shown in Fig. 3.2. Sensor v_i sends its message containing the aggregation data $f(s_i)$ to one of its neighbor sensors. After communications among other sensors in the wireless sensor network, sensor v_i may receive from one of its neighbor sensors a new message encapsulating an aggregation data, e.g., $f(s_i, s_j, s_k)$, which has aggregated the measurement data s_i it transmitted to its neighbor sensors previously. The data aggregation at sensor v_i will then result in bias in the aggregation data shown as $f(s_i, s_i, s_j, s_k)$ by the dashed line at sensor v_i.

In these two situations, the bias happens at the sensor that performs the data aggregation since this sensor does not know of the existence of the same measurement data in the aggregation data from other sensors.

The I-Header contains the information to identify the existence of the measurement data in the aggregation data of several sensors. Therefore, when the corresponding bits are one in the I-Headers of more than one incoming messages of the sensor, there will be bias when the sensor performs the data aggregation.

3.2.2 Multiset expression

To facilitate the expression of the data aggregation which yields bias, the concept of multiset can be applied [KLT03] since it enables the existence of an element more than once. Furthermore, the multiset can be combined with I-Header to enable the bias reduction introduced later.

Using the concept of multiset, the set of measurement data aggregated in the aggregation data at sensor v_i can be denoted by $(\mathcal{S}_i^{\mathrm{i}}, c_{\mathcal{S}_i})$, where the first element $\mathcal{S}_i^{\mathrm{i}}$ is so-called *underlying set* of the multiset and the second element is the *multiplicity* which is a function $c_{\mathcal{S}_i} : \mathcal{S}_i^{\mathrm{i}} \to \mathbb{N}_{\geq 1}$ mapping each element in $\mathcal{S}_i^{\mathrm{i}}$ to a non-zero integer value. If measurement data s_j of sensor v_j is aggregated in the aggregation data at sensor v_i, i.e., $s_i \in \mathcal{S}_i$, index j is contained in the index set $\mathcal{S}_i^{\mathrm{i}}$, i.e., $i \in \mathcal{S}_i^{\mathrm{i}}$. Meanwhile, $c_{\mathcal{S}_i}(j) \geq 1$ indicates how many times data s_j has been aggregated in the aggregation data at sensor v_i.

To quantify the bias, we now focus on a sensor v_i and its neighbor sensors $v_j \in \mathcal{N}_i$. Let m_i denote the message that is to be communicated by sensor $v_i \in \mathcal{V}$. Let $\mathcal{N}_i^{\mathrm{S}} \subseteq \mathcal{N}_i$ denote a subset of the neighbor sensors of sensor v_i which are intended to transmit their messages $m_j, v_j \in \mathcal{N}_i^{\mathrm{S}}$ to v_i. The corresponding set of measurement data in the aggregation data encapsulated in message m_j is $\mathcal{S}_j$, and the index set is $\mathcal{S}_j^{\mathrm{i}}$. When

sensor v_i receives the messages of every sensor in $\mathcal{N}_i^{\mathrm{S}}$, and it performs data aggregation to the aggregation data encapsulated in these messages, the set of measurement data in the output aggregation data is $\mathcal{S}_i^{\mathrm{i}'}$. The set of measurement data $\mathcal{S}_i^{\mathrm{i}'}$ is the accumulation of all sets of measurement data $\mathcal{S}_j, v_j \in \mathcal{N}_i^{\mathrm{S}}$ and the set of measurement data $\mathcal{S}_i$ of sensor v_i. Let m_i' denote the new message that encapsulates the output aggregation data. The underlying set of the aggregation data in the message m_i' is

$$\mathcal{S}_i^{\mathrm{i}'} = \mathcal{S}_i^{\mathrm{i}} \cup (\cup_{v_j \in \mathcal{N}_i^{\mathrm{S}}} \mathcal{S}_j^{\mathrm{i}}) \tag{3.1}$$

and the multiplicity $c_{\mathcal{S}_i'}$ of a data index $l \in \mathcal{S}_i^{\mathrm{i}'}$ is

$$c_{\mathcal{S}_i'}(l) = c_{\mathcal{S}_i}(l) + \sum_{v_j \in \mathcal{N}_i^{\mathrm{S}}} c_{\mathcal{S}_j}(l). \tag{3.2}$$

It shall be noticed that if the measurement data, e.g., s_k, is not in a set $\mathcal{S}_l$ but is in another set $\mathcal{S}_k$, the summation in the multiplicity yields $c_{\mathcal{S}_l}(k) + c_{\mathcal{S}_k}(k) = c_{\mathcal{S}_k}(k)$.

If the index multiplicity $c_{\mathcal{S}_i'}(m)$ of the measurement data s_m in $\mathcal{S}_i'$ is greater than 1, the bias of s_m is

$$b_{m_i'}(s_m) = c_{\mathcal{S}_i'}(m) - 1. \tag{3.3}$$

We quantify the bias $b_{m_i'}$ of the aggregation data encapsulated in message m_i' by

$$b_{m_i'} = \sum_{l \in \mathcal{S}_i^{\mathrm{i}'}} b_{m_i'}(s_m) = \sum_{l \in \mathcal{S}_i^{\mathrm{i}'}} (c_{\mathcal{S}_i'}(l) - 1). \tag{3.4}$$

3.3 Principle of bias reduction

The previous section defines the bias of the measurement data and the bias of aggregation data. The detection of the bias can be enabled by applying Θ function to I-Headers introduced in Chapter 2 to get the index set. In this section, the principle of the bias cancellation and how to perform it at the sensor in wireless sensor networks will be discussed.

Firstly, the detection of the bias is introduced in the context of the divisible functions discussed in Chapter 2.

When sensor v_i receives a message from sensor v_j, the set of the measurement data aggregated in the aggregation data encapsulated in the message is $\mathcal{S}_j$, and the corresponding index set is $\mathcal{S}_j^{\mathrm{i}}$. The set of measurements in the aggregation data at sensor

v_i is $\mathcal{S}_i$, and the index set is $\mathcal{S}_i^{\mathrm{i}}$. It is assumed that there is no bias of measurement data in $\mathcal{S}_i$ and $\mathcal{S}_j$ so far, the cardinalities of $\mathcal{S}_i$ and $\mathcal{S}_j$ are l_i and l_j, respectively. Let the data set $\mathcal{S}_{ij}^{\mathrm{B}}$ denote the intersection of sets $\mathcal{S}_i$ and $\mathcal{S}_j$, i.e.,

$$\mathcal{S}_{ij}^{\mathrm{B}} = \mathcal{S}_i \cap \mathcal{S}_j. \tag{3.5}$$

If $\mathcal{S}_{ij}^{\mathrm{B}}$ is not an empty set, i.e., $\mathcal{S}_{ij}^{\mathrm{B}} \neq \phi$, the aggregation at sensor v_i results in bias. Hence, in the divisible function

$$f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}, \boldsymbol{s}_{\mathcal{S}_j}) = g^{\Pi(\{\mathcal{S}_i,\mathcal{S}_j\})}\left(f_{l_i}(\boldsymbol{s}_{\mathcal{S}_i}), f_{l_j}(\boldsymbol{s}_{\mathcal{S}_j})\right) \ , \tag{3.6}$$

there is measurement data being aggregated more than once.

Intuitively, in order to reduce the bias, the measurement data that has been aggregated more than once has to be subtracted from the computation in (3.6). However, the measurement data may not be available at a sensor in the form that it can be used to subtract the bias directly from the aggregation data with bias. Quite the contrary, the measurement data of bias may have been aggregated in some aggregation data together with other measurement data. In the following, a method is proposed to combine several aggregation data in order to subtract the measurement data for the bias reduction.

We assume there are some aggregation data whose corresponding sets of measurement data are $\mathcal{S}_1^{v_i}, \mathcal{S}_2^{v_i}, \cdots$. The superscript v_i indicates that all the sets are available at sensor v_i. The availability is a result of the communication of sensor v_i and its neighbor sensors. Let a set $\Psi^{v_i} = \{\mathcal{S}_1^{v_i}, \mathcal{S}_2^{v_i}, \cdots, \mathcal{S}_{\psi_i}^{v_i}\}$ collect ψ_i sets of measurement data which are available at v_i, where ψ_i is the number of data sets in Ψ^{v_i}. The corresponding data vectors of the sets of measurement data included in Ψ^{v_i} are denoted by $\boldsymbol{s}_{\mathcal{S}_1^{v_i}}, \boldsymbol{s}_{\mathcal{S}_2^{v_i}}, \cdots \boldsymbol{s}_{\mathcal{S}_{\psi_i}^{v_i}}$. The aggregation data that outputs from aggregating the measurement data in each set of Ψ^{v_i} are then denoted by $f_{l_1^{v_i}}(\boldsymbol{s}_{\mathcal{S}_1^{v_i}}), f_{l_2^{v_i}}(\boldsymbol{s}_{\mathcal{S}_2^{v_i}}), \cdots, f_{l_{\phi_i}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{\psi_i}^{v_i}})$.

Let Υ denote a multiset operation which is either the union of two sets, $\cup$, or the set-theoretic difference $\setminus$. Applying the operations to all the sets of measurement data given in set Ψ^{v_i} results in

$$\mathcal{S}_1^{v_i}\Upsilon_1\mathcal{S}_2^{v_i}\Upsilon_2\cdots\Upsilon_{\psi_i-1}\mathcal{S}_{\psi_i}^{v_i} = \mathcal{S}_{ij}^{\mathrm{B}} \ , \tag{3.7}$$

where the operation is Υ_i between the set $\mathcal{S}_i^{v_i}$ and the set $\mathcal{S}_{i+1}^{v_i}$. The result given by (3.7) considers all possible combinations to sets $\mathcal{S}_1^{v_i}$, $\mathcal{S}_2^{v_i}$, ..., $\mathcal{S}_{\psi_i}^{v_i}$.

Let $\mathcal{S}_{1\to i}^{v_i}$ denote the accumulated results from $\mathcal{S}_1^{v_i}$ to $\mathcal{S}_i^{v_i}$, i.e. $\psi_i = 1$ in (3.7), correspondingly, let $\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}$ be the accumulated data vector and $f_{l_{1\to i}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}})$ be the aggregation date.

There are then two possible operations:

- When the operation Υ_i is a union $\cup$, the corresponding operation applied to the aggregation data is

$$f_{l_{1\to i}^{v_i}+l_{i+1}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}, \boldsymbol{s}_{\mathcal{S}_{i+1}^{v_i}}) = g^{\Pi(\{\mathcal{S}_{1\to i}^{v_i}, \mathcal{S}_{i+1}^{v_i}\})}\left(f_{l_{1\to i}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}), f_{l_{l+1}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{l+1}^{v_i}})\right) . \tag{3.8}$$

  It shall be noted that there could be duplications of measurement data in the operations.

- When the operation Υ_i is a set-theoretic difference $\setminus$, the corresponding operation applied to the aggregation output is shall only be applied under two conditions:
  - All data contained in the set $\mathcal{S}_{i+1}^{v_i}$ is contained in $\mathcal{S}_{1\to i}^{v_i}$, and
  - there exists an inverse function $g^{-\Pi(\{\mathcal{S}_{1\to i}^{v_i}, \mathcal{S}_{i+1}^{v_i}\})}$ which takes $f_{l_{1\to i}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}})$ and $f_{l_{l+1}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{l+1}^{v_i}})$ as input parameters and yields an aggregation with the data in the data set $\mathcal{S}_{1\to i}^{v_i} \setminus \mathcal{S}_{l+1}^{v_i}$.

  When both conditions are fulfilled, the aggregation data output from the operation is

$$f_{l_{1\to i}^{v_i}-l_{i+1}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}, \boldsymbol{s}_{\mathcal{S}_{i+1}^{v_i}}) = g^{-\Pi(\{\mathcal{S}_{1\to i}^{v_i}, \mathcal{S}_{i+1}^{v_i}\})}\left(f_{l_{1\to i}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}), f_{l_{l+1}^{v_i}}(\boldsymbol{s}_{\mathcal{S}_{l+1}^{v_i}})\right) . \tag{3.9}$$

  If the conditions resulting in a valid corresponding set-theoretical difference are not fulfilled, the given combination of the set of measurement data and the operations are then not considered in the bias cancellation.

After applying the operations Υ to the data set in Ψ^{v_i}, the corresponding aggregation output gives $f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})$. To reduce the bias in the computation (3.6), one can simply apply

$$f_{l_i+l_j-l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{UB}}}) = g^{-\Pi(\{\{\mathcal{S}_i, \mathcal{S}_j\}, \mathcal{S}_{ij}^{\mathrm{B}}\})}(f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}, \boldsymbol{s}_{\mathcal{S}_j}), f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})) , \tag{3.10}$$

where the set of measurement date is $\mathcal{S}_{ij}^{\mathrm{UB}} = \mathcal{S}_i \cup \mathcal{S}_j$, and the superscript UB implies that it is an UnBiased version after the bias of the measurement data included in $\mathcal{S}_{ij}^{\mathrm{B}}$

is eliminated by the computation in (3.10). $\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{UB}}}$ is the accumulated data vector of the measurement data in $\mathcal{S}_{ij}^{\mathrm{UB}}$. The cardinality of $\mathcal{S}_{ij}^{\mathrm{UB}}$ is denoted by l_{ij}^{UB} which is equal to $l_i + l_j - l_{ij}^{\mathrm{B}}$.

We provide a toy example to demonstrate the operations in (3.7). Assuming that the data set of the current message at sensor v_i is $\mathcal{S}_i = \{s_1, s_2, s_3, s_4\}$, the data set of the incoming message from sensor v_j is $\mathcal{S}_j = \{s_3, s_4, s_5, s_6\}$, the set Ψ^{v_i} contains four data sets, $\mathcal{S}_1^{v_i} = \{s_1, s_2, s_3, s_4\}$, $\mathcal{S}_2^{v_i} = \{s_1, s_2, s_4\}$, $\mathcal{S}_3^{v_i} = \{s_2, s_4\}$ and $\mathcal{S}_4^{v_i} = \{s_4\}$ and the data set $\mathcal{S}_{ij}^{\mathrm{B}} = \mathcal{S}_i \cap \mathcal{S}_j = \{s_3, s_4\}$. Then the set of operations which are applied to $\mathcal{S}_1^{v_i}$, $\mathcal{S}_2^{v_i}$, $\mathcal{S}_3^{v_i}$ and $\mathcal{S}_4^{v_i}$ is

$$\mathcal{S}_1^{v_i} \setminus \mathcal{S}_2^{v_i} \cup \mathcal{S}_3^{v_i} \setminus \mathcal{S}_4^{v_i} = \mathcal{S}_{ij}^{\mathrm{B}} .$$

In Chapter 2, we list some examples of the divisible functions. When there exists duplication of data, not all the functions require a set Ψ^{v_i} and perform the bias-cancellation stated in (3.7). It is because the duplication of measurement data does not impact the computation result. For example, the max function $f_N(\boldsymbol{s}) = \max_i s_i$ and the min function $f_N(\boldsymbol{s}) = \min_i s_i$ are not influenced by the bias because taking the max/min from a data set $\mathcal{S}_i$ is always equivalent to taking the max/min from the data set $\mathcal{S}_i \cup \{s_j\}$ when $s_j \in \mathcal{S}_i$.

Other divisible functions such as downloading, histogram, sum, and average functions will suffer from the duplication of measurement data. In order to perform the bias-cancellation in (3.7) and its corresponding operations on the aggregation output, it needs to be tested against the existence of an inverse function $g^{-\Pi}$ in order to apply the equation for bias cancellation (3.10).

- Downloading function: the computation in (3.10) is

$$\begin{aligned} & g^{-\Pi(\{\{\mathcal{S}_i,\mathcal{S}_j\},\mathcal{S}_{ij}^{\mathrm{B}}\})}(f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}, \boldsymbol{s}_{\mathcal{S}_j}), f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})) \qquad (3.11) \\ = \;& \text{delete } \boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}} \text{ from } \boldsymbol{s}_{\mathcal{S}_{ij}}. \end{aligned}$$

- Histogram function: the computation in (3.10) is

$$\begin{aligned} & g^{-\Pi(\{\{\mathcal{S}_i,\mathcal{S}_j\},\mathcal{S}_{ij}^{\mathrm{B}}\})}(f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}, \boldsymbol{s}_{\mathcal{S}_j}), f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})) \qquad (3.12) \\ = \;& f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}) - f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}}). \end{aligned}$$

- Sum function: the computation in (3.10) is

$$\begin{aligned} & g^{-\Pi(\{\{\mathcal{S}_i,\mathcal{S}_j\},\mathcal{S}_{ij}^{\mathrm{B}}\})}(f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}, \boldsymbol{s}_{\mathcal{S}_j}), f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})) \qquad (3.13) \\ = \;& f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i}) - f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}}). \end{aligned}$$

- Average function: the computation in (3.10) is

$$\begin{aligned} & g^{-\Pi(\{\{\mathcal{S}_i,\mathcal{S}_j\},\mathcal{S}_{ij}^{\mathrm{B}}\})}(f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i},\boldsymbol{s}_{\mathcal{S}_j}),f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})) \\ = & \frac{(l_i+l_j)f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i})-l_{ij}^{\mathrm{B}}f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})}{l_i+l_j-l_{ij}^{\mathrm{B}}}. \end{aligned} \tag{3.14}$$

As shown above, to perform bias reduction consists of two steps. The first is to determine the bias $\mathcal{S}_{ij}^{\mathrm{B}}$, and the second is to perform the Υ operation to several sets of measurement data collected in set Ψ^{v_i}. Equivalently, for the first one, one can find the bias in the form of the index set $\mathcal{S}_{ij}^{\mathrm{iB}}=\mathcal{S}_i^{\mathrm{i}}\cap\mathcal{S}_j^{\mathrm{i}}$ since the measurement data cannot be explicitly retrieved and is always computed in aggregation data. For the second one, the measurement data in each set $\mathcal{S}_i^{v_i}\in\Psi^{v_i}$ is aggregated in the aggregation data encapsulated in a message which is, together with the I-Header, available at sensor v_i. Therefore, the conditions of applying the bias cancellation shown in (3.10) are

- sensor v_i knows the I-Header of its own message m_i and the message m_j from sensor v_j,
- sensor v_i knows messages where the data set $\mathcal{S}_i^{v_i}\in\Psi^{v_i}$ is aggregated, and their corresponding I-Headers,
- sensor v_i knows a set of operations Υ which fulfills (3.7).

Based on the principle of the method mentioned above, a bias-cancellation algorithm is proposed as shown in Algorithm 1.

Algorithm 1 Bias cancellation algorithm

1: Sensor v_j sends its I-Header $\boldsymbol{I}_j$ and its message to sensor v_i.
2: v_i gets the index sets $\mathcal{S}_i^{\mathrm{i}}$ and $\mathcal{S}_j^{\mathrm{i}}$ by applying $\Theta(\boldsymbol{I}_i)$ and $\Theta(\boldsymbol{I}_j)$, respectively.
3: The indices of the data that leads to bias are $\mathcal{S}_{ij}^{\mathrm{iB}}=\mathcal{S}_i^{\mathrm{i}}\cap\mathcal{S}_j^{\mathrm{i}}$.
4: v_i finds messages which data in data set $\mathcal{S}_i^{v_i}\in\Psi^{v_i}$ is aggregated and finds the set of operations Υ using exhaust search.
5: v_i computes $f_{l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{B}}})$.
6: v_i computes $f_{l_i+l_j}(\boldsymbol{s}_{\mathcal{S}_i},\boldsymbol{s}_{\mathcal{S}_j})$.
7: v_i computes $f_{l_i+l_j-l_{ij}^{\mathrm{B}}}(\boldsymbol{s}_{\mathcal{S}_{ij}^{\mathrm{UB}}})$ using (3.10).

3.4 Sensor selection in bias reduction

3.4.1 Introduction

Bias reduction can be jointly considered with sensor selection when there are more than one neighbor sensors transmitting messages to a sensor.

Figure 3.3 considers a sensor v_i and its neighbor sensors where sensors v_j, v_k, v_l and v_m transmit their messages to sensor v_i successfully.

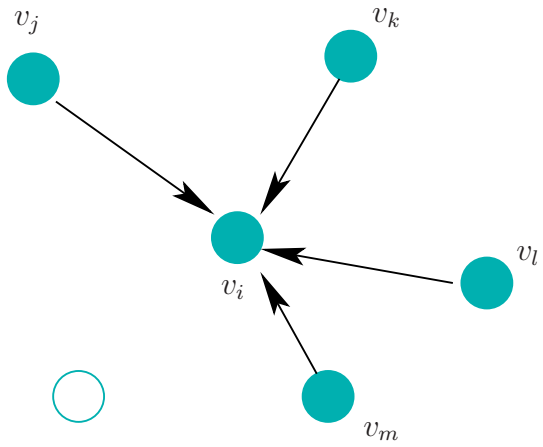

Figure 3.3. v_i receives messages from more than one neighbor sensor

A straightforward extension of the bias cancellation introduced in the previous section is to iteratively cancel the bias in the aggregation data after the aggregation with the data in each incoming message. However, this may lead to a problem of redundant message transmission. We consider here two examples using Figure 3.3, where sensor v_i receives messages from v_j, v_k, v_l, and v_m consecutively.

- In the first example, the set of measurement data in the aggregation data of sensor v_j and that of sensor v_k fulfill $\mathcal{S}_j \supset \mathcal{S}_k$, the message transmission from sensor v_k is not necessary.
- In the second example, the corresponding data sets have the relation $\mathcal{S}_j \cup \mathcal{S}_k \subseteq \mathcal{S}_l$, i.e., the measurement data aggregated in the message from sensor v_l includes all measurement data that is transmitted from sensor v_j and v_k, the message transmission of sensors v_j and v_k will be redundant.

In this section, a two-step solution is proposed to achieve the optimum performance for bias cancellation when multiple neighbor sensors of sensor v_i are intended to transmit their messages to v_i. The first step is to perform a selection of neighbor sensors of sensor v_i in order to find a subset of $\mathcal{N}_i^{\mathrm{S}}$. The aggregation data of the selected sensors will be taken into account in the data aggregation and the bias reduction at sensor v_i. The second step is to perform the bias reduction to the data aggregation at sensor v_i. The data aggregation considers the aggregation data from the selected neighbor sensors and the aggregation data of sensor v_i.

3.4.2 Selection of neighbor sensors

In the first step, we will solve the problem of selecting a subset of the neighbor sensors whose aggregation data are considered in the data aggregation at sensor v_i. A constraint of the solution is that all of the measurement data aggregated in the aggregation data encapsulated in the message of neighbor sensors of v_i shall be included after the selection.

Let $\mathcal{S}_i^{\mathrm{R}}$ denote the underlying data set if the measurement data of all the data sets $\mathcal{S}_j, v_j \in \mathcal{N}_i^{\mathrm{S}}$ are aggregated at sensor v_i, i.e.,

$$\mathcal{S}_i^{\mathrm{R}} = \mathcal{S}_i \cup \left(\cup_{v_j \in \mathcal{N}_i^{\mathrm{S}}} \mathcal{S}_j\right) , \tag{3.15}$$

where $\mathcal{S}_i^{\mathrm{R}}$ is called *the reference data set of aggregation* at sensor v_i and as mentioned in previous sections that if the measurement data of a sensor is in the underlying data set, it only appears once. To facilitate the bias reduction algorithm which is going to be introduced, $\mathcal{S}_i^{\mathrm{R}}$ is represented using multiset as $(\mathcal{S}_i^{\mathrm{R}}, \mathbf{1})$, where $\mathbf{1}$ is an all-1-vector with the length being the cardinality of $\mathcal{S}_i^{\mathrm{R}}$. The data set $\mathcal{S}_i^{\mathrm{R}}$ includes all measurement data that shall be aggregated when sensor v_i receives messages from its neighbor sensor regardless of the neighbor sensors selection. Therefore, $\mathcal{S}_i^{\mathrm{R}}$ gives the reference in such a way that when sensor v_i selects its neighbor sensors, the measurement data that are aggregated in the aggregation data encapsulated in the messages of the selected neighbors shall include all measurement data given in $\mathcal{S}_i^{\mathrm{R}}$ and sensor v_i will observe the complete set of measurement from its neighbor sensors.

It can be summarized from the two examples of where redundant messages are transmitted, to select the neighbor sensors, the set of measurement data that are aggregated in the aggregation data need to be compared. For simplicity, we set up a dictionary of the comparison results between two sets of measurement data.

Let r^{Θ} be a function with two data sets $\mathcal{S}_i$ and $\mathcal{S}_j$ as parameters and output an index indicating the relation between two data sets:

$$r^{\Theta}(\mathcal{S}_i, \mathcal{S}_j) = \begin{cases} 1 & \text{for } \mathcal{S}_i = \mathcal{S}_j \ , \\ 2 & \text{for } \mathcal{S}_i \supset \mathcal{S}_j \ , \\ 3 & \text{for } \mathcal{S}_i \subset \mathcal{S}_j \ , \\ 4 & \text{else} \ . \end{cases} \tag{3.16}$$

When the result is 4, it indicates that both of the two data sets have new data to each other.

The collection of all sets of measurement data of the messages at sensors in $\mathcal{N}_i^{\mathrm{S}}$ is denoted by an ordered set $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ with $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}} = \{\mathcal{S}_j | v_j \in \mathcal{N}_i^{\mathrm{S}}\}$. Let the sets of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ be ordered according to the increase of the indices of the corresponding sensors such that $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}(l), l = 1, 2, \cdots, |\mathcal{N}_i^{\mathrm{S}}|$ can be used to denote the l-th set of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ which is from the l-th sensor in $\mathcal{N}_i^{\mathrm{S}}$.

The proposed selection method of the neighbor sensors is based on the grouping. All sets of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ will be grouped according to the comparison results between the sets of measurement data.

Let $\mathcal{P}$ denote the set of groups generated by grouping the sets of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ and let p denote the number of groups in $\mathcal{P}$ after grouping. Let $\mathcal{P}_j, j = 1, 2, \cdots, p$ denote the j-th group in $\mathcal{P}$. In group $\mathcal{P}_j$, let variable $n_{\mathcal{P}_j}$ denote the number of sets of measurement data and let $\mathcal{P}_j(l), l = 1, 2, \cdots, n_{\mathcal{P}_j}$ denote the l-th set of measurement data in $\mathcal{P}_j$. The sets of measurement data in each group in $\mathcal{P}$ are ordered such that the first set of a group is a superset of all other sets of measurement data in the same group. The first step of grouping is to find the first set of measurement data of every group in $\mathcal{P}$. Let $\mathcal{P}^1$ denote the set collecting the first sets of measurement data of all groups in $\mathcal{P}$, i.e., $\mathcal{P}^1 = \{\mathcal{P}_1(1), \mathcal{P}_2(1), \cdots, \mathcal{P}_p(1)\}$. The Algorithm 2 generates the set $\mathcal{P}^1$ given the input of $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$. In Algorithm 2, whenever a set of measurement data contains new measurement data that is not included in other sets of measurement data which are already in $\mathcal{P}^1$, the set will be included in $\mathcal{P}^1$. Meanwhile, if a data set in $\mathcal{P}^1$ is tested as a subset of another data set, it will be eliminated from $\mathcal{P}^1$.

Algorithm 2 yields the set $\mathcal{P}^1$ as well as the number p of groups being the number of data sets in $\mathcal{P}^1$. Since every set of measurement data which contains new measurement data is included in $\mathcal{P}^1$, the union of all data sets in $\mathcal{P}^1$ is identical to $\mathcal{S}_i^{\mathrm{R}}$ satisfying

$$\cup_{\mathcal{S}_j \in \mathcal{P}^1} \mathcal{S}_j = \mathcal{S}_i^{\mathrm{R}} \ . \tag{3.17}$$

Algorithm 2 Algorithm to find $\mathcal{P}^1$

$\mathcal{P}^1$ is initialized to be $\mathcal{P}^1 = \{\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}(1)\}$
for $\mathcal{S}_j$ in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$ **do**
  join_$\mathcal{P}^1 := 0$;
  for $\mathcal{S}_k$ in $\mathcal{P}^1$ **do**
    if $r^{\Theta}(\mathcal{S}_j, \mathcal{S}_k) = 1 \text{or} = 3$ **then**
      join_$\mathcal{P}^1 := 0$;
      Stop current **for**-loop and start with the next $\mathcal{S}_j$;
    else
      if $r^{\Theta}(\mathcal{S}_j, \mathcal{S}_k) \in \{2\}$ **then**
        join_$\mathcal{P}^1 := 1$;
        $\mathcal{P}^1 = \mathcal{P}^1 \setminus \mathcal{S}_k$;
      else
        join_$\mathcal{P}^1 := 1$;
      end if
    end if
  end for
  if join_$\mathcal{P}^1 = 1$ **then**
    $\mathcal{P}^1 = \mathcal{P}^1 \cup \{\mathcal{S}_j\}$;
  end if
end for

The second step of the grouping is to assign the sets of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{s}}}$ to the p groups using the algorithm shown in Algorithm 3 with the knowledge of $\mathcal{P}^1$. It shall be noticed that every data set in $\mathcal{P}^1$ can be assigned to only one of the p groups. However, the other data sets in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}} \setminus \mathcal{P}^1$ can be assigned to more than one group.

For example, the data sets in $\Psi_i^{\mathcal{N}_i^{\mathrm{s}}}$ are $\{\mathcal{S}_1, \cdots, \mathcal{S}_6\}$, where $\mathcal{S}_1 = \{s_1, s_2, s_3\}$, $\mathcal{S}_2 = \{s_2, s_3, s_4\}$, $\mathcal{S}_3 = \{s_4, s_5\}$, $\mathcal{S}_4 = \{s_1, s_2\}$, $\mathcal{S}_5 = \{s_2\}$ and $\mathcal{S}_6 = \{s_3, s_4\}$, respectively. The reference data set of aggregation is $\mathcal{S}_i^{\mathrm{R}} = \{s_1, s_2, s_3, s_4, s_5\}$. Set $\mathcal{P}^1$ is $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$ according to Algorithm 2 and the result of the grouping by using algorithm in Algorithm 3 is $\mathcal{P}_1 = \{\mathcal{S}_1, \mathcal{S}_4, \mathcal{S}_5\}$, $\mathcal{P}_2 = \{\mathcal{S}_2, \mathcal{S}_5, \mathcal{S}_6\}$ and $\mathcal{P}_3 = \{\mathcal{S}_3\}$.

After grouping of the data sets, one data set will be chosen from each group in $\mathcal{P}$ together with the data set $\mathcal{S}_i$ to test whether their union is equivalent to $\mathcal{S}_i^{\mathrm{R}}$. If the equivalence holds, the current selection data sets is a candidate for calculating the function output, and the bias cancellation is performed based on the selected data set. Let set $\mathcal{C}$ collect all these possible selections of the data sets, and let $n_{\mathcal{C}}$ denote the number of possible selections collected in $\mathcal{C}$ where each selection has p data sets chosen from each group of $\mathcal{P}$. Let set $\mathcal{C}_m, m = 1, 2, \cdots, n_{\mathcal{C}}$ denote the m-th selection in $\mathcal{C}$ and $\mathcal{C}_m(l), l = 1, 2, \cdots, p$ denote the l-th data set in $\mathcal{C}_m$. With the example above, we have $\mathcal{C} = \{\mathcal{C}_1, \cdots, \mathcal{C}_4\}$, where $\mathcal{C}_1 = \{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$, $\mathcal{C}_2 = \{\mathcal{S}_1, \mathcal{S}_6, \mathcal{S}_3\}$, $\mathcal{C}_3 = \{\mathcal{S}_4, \mathcal{S}_2, \mathcal{S}_3\}$ and

Algorithm 3 Algorithm for partitioning and grouping

1: **for** $\mathcal{S}_k$ in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}} \setminus \mathcal{P}^1$ **do**
2: **for** $\mathcal{P}_j$ in $\mathcal{P}$ **do**
3: join_group := 0;
4: join_group_END := 1;
5: **for** $l = 1$ **to** $n_{\mathcal{P}_j}$ **do**
6: **if** $r^{\Theta}(\mathcal{S}_k, \mathcal{P}_j(l)) \in \{1\}$ **then**
7: join_group := 0;
8: Stop the current **for**-loop and go to line 21
9: **else**
10: **if** $r^{\Theta}(\mathcal{S}_k, \mathcal{P}_j(l)) \in \{3\}$ **then**
11: join_group := 1;
12: **else**
13: **if** $r^{\Theta}(\mathcal{S}_k, \mathcal{P}_j(l)) \in \{2\}$ **then**
14: join_group := 1;
15: join_group_end := 0;
16: Stop the current **for**-loop and go to line 21
17: **end if**
18: **end if**
19: **end if**
20: **end for**
21: **if** join_group = 1 **then**
22: **if** join_group_END = 0 **then**
23: $\mathcal{S}_k$ joins the group $\mathcal{P}_j$ at the last position, i.e., $\mathcal{P}_j(n_{\mathcal{P}_j} + 1) = \mathcal{S}_k$;
24: **else**
25: $\mathcal{S}_k$ joins the group $\mathcal{P}_j$ at the l-th position;
26: **end if**
27: **end if**
28: **end for**
29: **end for**

$\mathcal{C}_4 = \{\mathcal{S}_4, \mathcal{S}_6, \mathcal{S}_3\}$, respectively.

Data aggregation of the set of the data sets in each selection will have a bias. Let Ψ be a selection in $\mathcal{C}$, i.e., $\Psi \in \mathcal{C}$. The set of sensors whose data sets are included in Ψ is denoted by $\mathcal{N}_i^{\Psi}$. Similar to (3.1) and (3.2), the underlying data set after v_i has aggregated all data sets in Ψ is $\mathcal{S}^{\Psi} = \mathcal{S}_i \cup (\cup_{v_j \in \mathcal{N}_i^{\Psi}} \mathcal{S}_j)$, and the multiplicity of the measurement data in $\mathcal{S}^{\Psi}$ is $c_{\mathcal{S}^{\Psi}}(l) = c_{\mathcal{S}_i}(l) + \sum_{v_j \in \mathcal{N}_i^{\Psi}} c_{\mathcal{S}_j}(l)$, where $l \in \mathcal{S}^{\mathrm{i}\Psi}$, and $\mathcal{S}^{\mathrm{i}\Psi}$ is the index set corresponding to $\mathcal{S}^{\Psi}$. The bias of the output is then $b_{m(\mathcal{S}^{\Psi})} = \sum_{l \in \mathcal{S}^{\mathrm{i}\Psi}} (c_{\mathcal{S}^{\Psi}}(l) - 1)$.

In this section, data sets from each neighbor sensor in $\mathcal{N}_i^{\mathrm{S}}$ are grouped to generate possible selections. To select the neighbor sensors for message transmissions, all the possible selections shall be tested against their performance of bias reduction given

bias $b_{m(\mathcal{S}^{\Psi})}$ before the bias reduction. This test will be explained in the subsequent subsection.

3.4.3 Bias reduction with messages in the memory

In this section, we discuss the acquisition of the set of data set Ψ^{v_i} used by sensor v_i when v_i performs bias cancellation.

As shown in Chapter 2, each sensor is equipped with memory capability to store various kinds of data, e.g., the previously transmitted or received messages. In order to perform the bias cancellation to reduce or eliminate the bias, we leverage the aspect to use the messages that are stored in the memory of the sensor.

Let ψ_i denote the number of messages stored in the buffer of sensor v_i. We denote the messages and their I-Headers stored in the buffer by $m_l^{v_i}$ and $\boldsymbol{I}_l^{v_i}$, respectively, where $l = 1, 2, \cdots, \psi_i$. The corresponding data sets of the stored messages are denoted by $\mathcal{S}_1^{v_i}, \mathcal{S}_2^{v_i}, \cdots, \mathcal{S}_{\psi_i}^{v_i}$, which can be determined by applying the function Θ^{-1} to the corresponding I-Header, i.e., $\mathcal{S}_l^{v_i} = \Theta^{-1}(\boldsymbol{I}_l^{v_i})$. Let $\Psi^{v_i} = \{\mathcal{S}_1^{v_i}, \mathcal{S}_2^{v_i}, \cdots, \mathcal{S}_{\psi_i}^{v_i}\}$ denote the set which collects these data sets. Generally, not all data sets in Ψ^{v_i} need to be taken to construct the set for bias reduction.

In the following definitions, all the set will be ordered sets in order to ensure the indices, which are expressed by integer numbers, can be permuted to output different combinations. Let $\Omega \subseteq \Psi^{v_i}$ be a set of data sets. Let $\Omega(i)$ be the i-th data set in Ω, where $i = 1, 2, \cdots, |\Omega|$. Let Λ_Ω be the set which collects all possible permutations of data sets in Ω. For a permutation $x \in \Lambda_\Omega$, where $x(j)$ gives the j-th index in the permutation, Ω^x gives the set where the data sets in Ω are ordered according to the permutation defined in x, i.e., $\Omega^x(j) = \Omega(x(j))$. For example, for the set $\Omega = \{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$, one has $\Lambda_\Omega = \{\{1,2,3\}, \{1,3,2\}, \{2,1,3\}, \{2,3,1\}, \{3,1,2\}, \{3,2,1\}\}$. If $x = \{3,2,1\}$, it results $\Omega^x = \{\mathcal{S}_3, \mathcal{S}_2, \mathcal{S}_1\}$.

For every Ω^x, a set Υ is defined to collect the operation between data sets $\Omega^x(i)$ and $\Omega^x(i+1)$ sequentially. The operation is only limited to two set operations, the union $\cup$ and the set-theoretic difference $\setminus$. Since there are $|\Omega^x|$ data sets in Ω^x, the number of operations in Υ is $|\Omega^x| - 1$. The result of the operation is a data set $\mathcal{S}^{\Omega^x} = \Omega^x(1)\Upsilon(1)\Omega^x(2)\Upsilon(2)\cdots\Upsilon(|\Omega^x|-1)\Omega^x(|\Omega^x|)$, where $\Upsilon(i)$ is the i-th operation in Υ. The collection of all possible choices of Υ is denoted by $\coprod$. When applied to bias-cancellation, admissibility of the operation Υ should be satisfied. The admissibility

Algorithm 4 Check Admissibility of Υ

$\mathcal{S}^{\Omega^x} = \Omega^x(1)$;
for $(i = 1; i < |\Omega^x|; i = i + 1)$ **do**
  if $\Upsilon(i)$ is $\cup$ **then**
    $\mathcal{S}^{\Omega^x} = \mathcal{S}^{\Omega^x} \cup \Omega^x(i + 1)$;
    $c_{\mathcal{S}^{\Omega^x}}(l) = c_{\mathcal{S}^{\Omega^x}}(l) + c_{\Omega^x(i+1)}(l)$, $\forall s_l \in \Omega^x(i + 1)$;
  else
    if $\Omega^x(i+1) \backslash \mathcal{S}^{\Omega^x}$ is an empty set ϕ **and** $c_{\mathcal{S}^{\Omega^x}}(l) - c_{\Omega^x(i+1)}(l) > 0, \forall s_l \in \Omega^x(i+1)$ **then**
      $\mathcal{S}^{\Omega^x} = \mathcal{S}^{\Omega^x} \setminus \Omega^x(i + 1)$;
      $c_{\mathcal{S}^{\Omega^x}}(l) = c_{\mathcal{S}^{\Omega^x}}(l) - c_{\Omega^x(i+1)}(l)$, $\forall s_l \in \Omega^x(i + 1)$;
    else
      $\mathcal{S}^{\Omega^x} = \phi$;
      STOP Current for-loop;
    end if
  end if
end for

holds when there is no operation $\Upsilon(i)$ subtracting a data resulting in a negative number of the data in the data set. Algorithm 4 can check the admissibility.

If the Algorithm 4 returns an empty set $\mathcal{S}^{\Omega^x} = \phi$, the failed admissibility results. Otherwise, the algorithm returns a multiset whose underlying data set is $\mathcal{S}^{\Omega^x}$, and the multiplicity is $c_{\mathcal{S}^{\Omega^x}}$. We define a *proper combination* of Ω as a tuple, which includes a permutation and a sequenced operation set (x, Υ). The remaining bias after performing bias cancellation using Ω and (x, Υ) is defined by

$$b(\Psi, \Omega, x, \Upsilon) = \sum_{l \in \mathcal{S}^{\mathrm{i}\Omega^x}} (c_{\mathcal{S}^{\Psi}}(l) - 1 - c_{\mathcal{S}^{\Omega^x}}(l)), \tag{3.18}$$

where $\mathcal{S}^{\mathrm{i}\Omega^x}$ is the index set of the data set $\mathcal{S}^{\Omega^x}$.

The goal of bias reduction at sensor v_i is to find the *bias-reduction set* $\Omega_{\mathrm{b}} \subseteq \Psi^{v_i}$, the subset of the data sets Ψ_{b} which includes in the message of the neighbor sensors of v_i, the optimal permutation x_{b} of the data sets in Ω_{b}, and the sequence of operations Υ_{b}

which all result from the minimization problem given by

$$
\begin{aligned}
&(\Psi_{\mathrm{b}}, \Omega_{\mathrm{b}}, x_{\mathrm{b}}, \Upsilon_{\mathrm{b}}) = \min_{\Psi,\Omega,x,\Upsilon} b(\Psi, \Omega, x, \Upsilon) \\
&\text{s.t.} \\
&\Psi \in \mathcal{C} \\
&c_{\mathcal{S}^{\Psi}}(l) - c_{\mathcal{S}^{\Omega^x}}(l) \geq 1, \forall s_l \in \mathcal{S}^{\Omega^x} \\
&\Omega \subseteq \Psi^{v_i} \\
&x \in \Lambda_{\Omega} \\
&\Upsilon \in \coprod .
\end{aligned}
\tag{3.19}
$$

The minimization problem in (3.19) yields a combinatorial problem. Firstly, its complexity analyzed. Assuming the number of data sets in Ω that is chosen from Ψ^{v_i} is no more than $\bar{\omega}$, i.e., $|\Omega| \leq \bar{\omega}$, the number of possible choices is

$$
\sum_{k=1}^{\bar{\omega}} \binom{\psi_i}{k} . \tag{3.20}
$$

For a given $1 < \omega \leq \bar{\omega}$, there will be $\omega - 1$ operations of Υ chosen for generating the bias-cancellation set. Considering the two choices of Υ, either being the union or being the set minus, the total number of possible choices of the operations for a chosen ω will be $2^{\omega-1}$. Therefore, there will be in total

$$
\sum_{\omega=2}^{\bar{\omega}-1} 2^{\omega-1} \tag{3.21}
$$

possible admissibility checks needed.

To reduce the time taken at the runtime of a sensor when it receives messages from its neighbor sensors, the number of computations that are needed to construct the bias reduction set shall be reduced. It can be easily observed that it requires no knowledge of the incoming messages at sensor v_i to perform admissibility checks. Therefore, the admissibility check of possible combinations of the messages in the memory can be done at sensors before the message communications start. Storing the output of all admissibility checks consumes memory of sensors, but it increases the speed of bias cancellation.

To limit the required memory to store the messages, we use ψ_i to indicate the maximum number of messages that sensor v_i can store in its memory. Furthermore, $\bar{\omega}$ is defined as *combination depth* to indicate the maximum number of data sets that can be combined

when performing computation. We define *the combination table* $\mathcal{T}$ as the output of admissibility checks.

To describe the algorithm of constructing $\mathcal{T}$, we denote $\Psi_{\omega}^{v_i}$ as the set of data sets resulting from chosen ω data sets from Ψ^{v_i} and $\Psi_{\omega}^{v_i}(l), l = 1, 2, \cdots, \omega$ is the l-th data set in $\Psi_{\omega}^{v_i}$. The table $\mathcal{T}$ consists of two parts. In the first part, for every ω and every possible set of data set $\Psi_{\omega}^{v_i}$, the union operations are applied. When the union operations are applied to the data sets in $\Psi_{\omega}^{v_i}$, the underlying data set of the result is denoted by $\mathcal{S}_{\omega}^{v_i}$, and the multiplicity is $c_{\mathcal{S}_{\omega}^{v_i}}(j) = \sum_{l=1}^{\omega} c_{\Psi_{\omega}^{v_i}(l)}(j)$. All of the outputs are inserted into $\mathcal{T}$. The algorithm to construct the first part of $\mathcal{T}$ is given in Algorithm 5.

Algorithm 5 Construction of the first part of table $\mathcal{T}$

```
for ω = 1 : ω̄ do
    for every possible Ψ_ω^{v_i} do
        determine the underlying data set S_ω^{v_i} and the multiplicity c_{S_ω^{v_i}}(j)
        T = T ∪ {S_ω^{v_i}}
    end for
end for
```

In the second part of $\mathcal{T}$, two data sets of the first part of $\mathcal{T}$ are chosen, and the set minus should be applied. However, the admissibility should be checked to determine whether the set minus operation can be applied. We denote $\mathcal{S}_1^{\mathcal{T}_1}$ and $\mathcal{S}_2^{\mathcal{T}_1}$ as the two ordered chosen data sets from the first part of $\mathcal{T}$. Their corresponding multiplicities are $c_{\mathcal{S}_1^{\mathcal{T}_1}}$ and $c_{\mathcal{S}_2^{\mathcal{T}_1}}$, respectively. The sets of data sets that are chosen from Ψ^{v_i} to construct $\mathcal{S}_1^{\mathcal{T}_1}$ and $\mathcal{S}_2^{\mathcal{T}_1}$ are $\Psi_{\mathcal{S}_1^{\mathcal{T}_1}}$ and $\Psi_{\mathcal{S}_2^{\mathcal{T}_1}}$, respectively.

Algorithm 6 Admissibility check for the construction of the second part of table $\mathcal{T}$

```
for every pair of S_1^{T_1} and S_2^{T_1} do
    if Ψ_{S_1^{T_1}} ∩ Ψ_{S_2^{T_1}} ≠ ψ then
        Admissibility check fails
    else
        if c_{S_1^{T_1}}(l) − c_{S_2^{T_1}}(l) < 0, ∀s_l ∈ S_2^{T_1} then
            Admissibility check fails
        else
            T = T ∪ {S_1^{T_1} \ S_2^{T_1}}
            c_{S_1^{T_1} \ S_2^{T_1}} = c_{S_1^{T_1}} − c_{S_2^{T_1}}
        end if
    end if
end for
```

The admissibility check given by Algorithm 6 fills $\mathcal{T}$ with the output of performing the set minus to the two chosen data sets from the first part of $\mathcal{T}$. After getting $\mathcal{T}$, it

is possible to select data set $\mathcal{S}^{\mathcal{T}}$ for performing the bias cancellation. We denote the multiplicity of the data in the chosen $\mathcal{S}^{\mathcal{T}}$ by $c_{\mathcal{S}^{\mathcal{T}}}$.

For better usage of the table $\mathcal{T}$, the bias of the aggregation output $\mathcal{S}^{\Psi}$ is counted using the name "layers". The number $L_{\mathcal{S}^{\Psi}}$ of layers is the maximum bias of the data in $\mathcal{S}^{\Psi}$, i.e.,

$$L_{\mathcal{S}^{\Psi}} = \max_{l} c_{\mathcal{S}^{\Psi}}(l) - 1 \ . \tag{3.22}$$

In the k-th layer $0 < k \leq L_{\mathcal{S}^{\Psi}}$, the corresponding bias of data s_l is

$$b^{k}_{m(\mathcal{S}^{\Psi})}(l) = 1, \tag{3.23}$$

if $c_{\mathcal{S}^{\Psi}}(l) - k > 0$. For the k-th layer, a data set $\mathcal{S}^{\mathcal{T}}_{k}$ is chosen from $\mathcal{T}$ to perform the bias cancellation. If there is bias remaining, it is accumulated in the next layer. Therefore, the bias cancellation problem can be simplified as the choice of the ordered data sets $\mathcal{S}^{\mathcal{T}}_{1}, \mathcal{S}^{\mathcal{T}}_{2} \cdots, \mathcal{S}^{\mathcal{T}}_{L_{\mathcal{S}^{\Psi}}}$ from $\mathcal{T}$, which leads to

$$\begin{aligned}
&(\Psi_{\mathrm{b}}, \mathcal{S}^{\mathcal{T}}_{1,b}, \cdots, \mathcal{S}^{\mathcal{T}}_{L_{\mathcal{S}^{\Psi}},b}) = \min_{\Psi, \mathcal{S}^{\mathcal{T}}_{1}, \cdots, \mathcal{S}^{\mathcal{T}}_{L_{\mathcal{S}^{\Psi}}}} b(\Psi, \mathcal{S}^{\mathcal{T}}_{1}, \cdots, \mathcal{S}^{\mathcal{T}}_{L_{\mathcal{S}^{\Psi}}}) \\
&\text{s.t.} \\
&\Psi \in \mathcal{C} \\
&\sum_{j=1}^{k} \left(b^{j}_{m(\mathcal{S}^{\Psi})}(l) - c_{\mathcal{S}^{\mathcal{T}}_{j}}(l) \right) \geq 0, \forall s_l \in \mathcal{S}^{\Psi} \ .
\end{aligned} \tag{3.24}$$

It can be deduced that the complexity of the bias cancellation given in (3.24) is dependent on the number $L_{\mathcal{S}^{\Psi}}$ of layers of the bias. Additionally, $L_{\mathcal{S}^{\Psi}}$ is relative to the number of neighbor sensors that are transmitting messages to sensor v_i.

3.5 Summary

In this chapter, the definition of bias using multiset expression has been introduced in the context of random gossiping. The bias cancellation has been proposed as an algorithm give by Algorithm 1. Furthermore, we have discussed the aspects to facilitate and improve the bias cancellation. The neighbor sensors selection helps to choose the neighbor sensors with the messages from whom the sensor performs aggregation. Using the messages in the sensor memory, we are able to find the optimal data set to perform the bias cancellation by solving an optimization problem. In this thesis, the optimization will be done numerically by exhaustive search. The performance will be evaluated using simulation in the next chapters.

Chapter 4

Aggregation time reduction

4.1 Introduction

The discussions in this chapter focus on the reduction of communications in random gossiping based wireless sensor networks.

As reviewed in Chapter 1, there are existing publications on the topic of improving convergence speed for consensus problem using topology optimization, i.e., tuning the connectivity between two sensors for every pair of sensors in the network. However, this method requires global information, and so far can only serve as an off-line solution. Moreover, this optimization is only for applying the consensus to all the sensor measurements.

Chapter 2 introduced a cross-layer model based on indicating-header (I-Header), which is shared information between the application layer and the network layer. I-Header can maintain the transparency of the two layers such that different applications can be supported. Based on this, Chapter 3 covers the bias cancellation topic.

In this chapter, we discuss the possibility of reducing the number of message communications in random gossiping based wireless sensor networks. The protocols of random gossiping using I-Headers for a dynamic wireless sensor network is introduced. Furthermore, we introduce multihop coordination in random gossiping as a method to further reduce the number of message communications.

The remainder of the chapter is organized as follows. In Section 4.2, we discuss the reduction of message communications in random gossiping in dynamic wireless sensor networks. Multihop coordination is discussed in Section 4.3. The performance of the proposed methods to reduce the aggregation time is given in Section 4.4. Section 4.5 concludes this chapter. Parts of the content of this chapter have been published by the author of this thesis in [CKK13b], [CKK13a] and [CKK14a].

4.2 Dynamic wireless sensor networks

4.2.1 Mobile sensors

We define sensors that may change their locations in wireless sensor networks as *mobile sensors*. A dynamic wireless sensor network shall meet two conditions:

- Sensors in the network are mobile sensors.
- The neighbor sensors of a sensor may change.

It shall be noticed that all sensors in the wireless sensor network being mobile sensors are not a sufficient condition of a dynamic wireless sensor network. If all sensors in the wireless sensor network are mobile sensors, but they are geographically relative stable when each sensor changes its location, the neighbor sensors of every sensor remain the same. In this case, the wireless sensor network is not dynamic.

In other words, a sensor does not have constant neighborhood information in dynamic wireless sensor networks. In this thesis, we assume that the sensors in a dynamic wireless sensor network change their location only after the communications of a sensor with its neighbor sensors being finished. Under this assumption, the sensor has a constant neighbor condition when it communicates with its neighbor sensors. The state machine in Figure 4.1 explains such process.

A sensor starts with neighbor discovery. Communications are only performed between sensors after neighbor sensors have been determined. A sensor may change its location after communications.

Depending on how a sensor is designated to interact with its neighbor sensors, we categorize sensors into humble sensors and greedy sensors, where the words *humble* and *greedy* indicate the number of neighbor sensors with which a sensor is going to exchange its message. When a sensor is a *humble sensor*, it exchanges messages with only one of its neighbor sensors, whereas a *greedy sensor* implies that a sensor will exchange its messages with all of its neighbor sensors. We call a sensor network a *humble wireless sensor network* when all the sensors in the network are humble sensors and a *greedy wireless sensor network* when all the sensors in the network are greedy sensors.

In the next two subsections, details of these two sensors and the corresponding random gossiping algorithms are discussed.

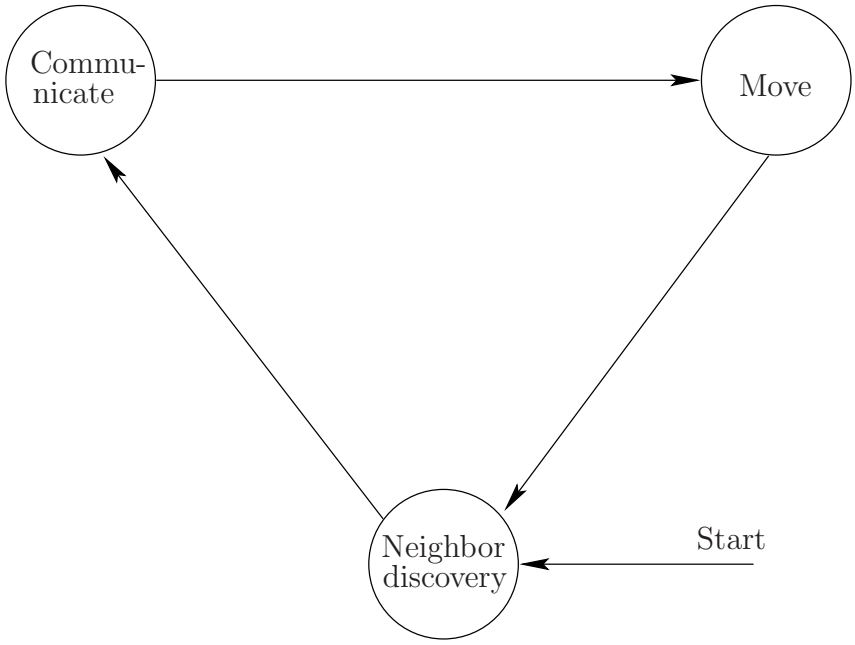

Figure 4.1. The state machine of sensors in dynamic wireless sensor networks

4.2.2 Random gossiping in humble wireless sensor networks

In humble wireless sensor networks, a sensor communicates its messages with only one of its neighbor sensors each time when it initializes the communications. The following problems need to be addressed according to the humble sensor definition.

- How does a sensor select one sensor from its neighbors to communicate?
- What is the characteristic of the network connectivity?
- What is the speed of convergence of the data aggregation in the network?

4.2.2.1 Neighbor sensor selection

For humble wireless sensor networks, we consider two ways to choose the neighbor sensor when a humble sensor initializes the communication.

The first way originates from the consensus problem, which is discussed in Chapter 2. The neighbor sensors are chosen randomly as proposed in [BGPS06]. With this neighbor selection method, one humble sensor v_i which is taken randomly from $\mathcal{V}$ initializes the communication at a time. The neighbor sensors of v_i are collected in the

set $\mathcal{N}_i$ with the cardinality N_i. When the selection of a sensor is equally probable, the probability $p_i(v_j)$ that sensor $v_j \in \mathcal{N}_i$ is chosen to perform the communication is

$$p_i(v_j) = \frac{1}{N_i}, \forall v_j \in \mathcal{N}_i \ . \tag{4.1}$$

The sensor selection in (4.1) chooses a neighbor sensor of v_i to exchange messages in a very simple way. However, it does not yield efficient communications. When sensor v_j is chosen randomly with probability $p_i(v_j)$, if sensor $v_j \in \mathcal{N}_i$ has the same set of data aggregated, it is of a probability $p_i(v_j$ that two redundant communications are performed. If either sensor v_i or sensor v_j has new data aggregated for the other, it is with a probability of $p_i(v_j)$ that one redundant communication is performed. If both v_i and v_j have new data for each other, we say there is no redundant communications has been performed. Generally, a sensor has no information on whether its neighbor sensor has aggregated specific measurement data. Therefore, the three cases have the same probability to happen, i.e., 1/3. The average number of redundant communications is

$$n^{\mathrm{r}} = \frac{1}{3} * 2 + \frac{1}{3} * 1 + \frac{1}{3} * 0 = 1 \ . \tag{4.2}$$

Meanwhile, if all sensors in $\mathcal{N}_i$ have the same distribution regarding the three cases, the average number of redundant communications when sensor v_i initializes the communications is

$$\sum_{v_j \in \mathcal{N}_i} p_i(v_j) * n^{\mathrm{r}} = 1. \tag{4.3}$$

In order to reduce the redundant transmission, we take advantage of I-Header that we introduced in Chapter 2. The I-Header of sensor v_i is $\boldsymbol{I}_i$, which can be translated into the corresponding data set $\mathcal{S}_i$ by the function $\mathcal{S}_i^{\mathrm{i}} = \Theta(\boldsymbol{I}_i)$. Before message transmission, sensor v_i broadcasts its $\boldsymbol{I}_i$. The neighbor sensors in $\mathcal{N}_i$ receives $\boldsymbol{I}_i$, perform the transform using $\Theta(\boldsymbol{I}_i)$ to get the index set $\mathcal{S}_i^{\mathrm{i}}$. After the comparison between $\mathcal{S}_i^{\mathrm{i}}$ and $\mathcal{S}_j^{\mathrm{i}}$ which is from $\boldsymbol{I}_j$ at sensor $v_j \in \mathcal{N}_i$, v_j can feed back an acknowledgment to sensor v_i containing its ID and the result after comparison indicating whether the communication is necessary. When sensor v_i receives that acknowledgment, a random neighbor sensor selection can still be performed, but in average one unnecessary transmission can be avoided since v_i or the selected neighbor can always choose to avoid the transmission based on the knowledge of the comparison.

The advantage of selecting a neighbor sensor randomly without using I-Header is its simplicity and its low overhead. The overhead required for two sensors, v_i and one of its neighbor sensors $v_j \in \mathcal{N}_i$ is merely their ID information and the overhead for synchronization. With the I-Header transmission, extra overhead includes the I-Header

transmission of sensor v_i and the acknowledgment from sensors in $\mathcal{N}_i$. The disadvantage is that it does not take into account the amount of measurement data that can be aggregated benefiting from communication. Therefore, we propose the neighbor selection using I-Header information.

When using I-Header, our objective to select a neighbor sensor is to maximize the mutual difference in the data set of sensor v_i and the selected neighbor sensor. The neighbor selection problem can be formulated as

$$v_j^{\mathrm{s}} = \arg\max_{v_j \in \mathcal{N}_i} | \left(\mathcal{S}_i^{\mathrm{i}} \cup \mathcal{S}_j^{\mathrm{i}}\right) \setminus \left(\mathcal{S}_i^{\mathrm{i}} \cap \mathcal{S}_j^{\mathrm{i}}\right) |, \tag{4.4}$$

where v_j^{s} is the selected sensor, and $|\cdot|$ returns the cardinality of the set. To implement the selection in (4.4), Algorithm 4.2.2.1 is proposed.

Algorithm 7 Neighbor selection for humble sensors with maximization of mutual difference

1: Sensor v_i firstly broadcasts its I-Header $\boldsymbol{I}_i$ to all sensors in $\mathcal{N}_i$
2: Neighbor sensors $v_j \in \mathcal{N}_i$ receives $\boldsymbol{I}_i$
3: v_j performs the translation using $S_i^{\mathrm{i}} = \Theta(\boldsymbol{I}_i)$
4: v_j computes $c_j = | \left(\mathcal{S}_i^{\mathrm{i}} \cup \mathcal{S}_j^{\mathrm{i}}\right) \setminus \left(\mathcal{S}_i^{\mathrm{i}} \cap \mathcal{S}_j^{\mathrm{i}}\right) |$
5: The computed output c_j is transmitted back to sensor v_i as an acknowledgement
6: Sensor v_i receives all acknowledgments from sensors in $\mathcal{N}_i$
7: The maximization in (4.4) is performed in order to select the sensor v_j^{s}.

4.2.2.2 Connectivity for humble wireless sensor networks

Connectivity in wireless sensor networks using random gossiping can be explained in a way that the data of one sensor will be aggregated at any other sensors with finite communication rounds. We verify the connectivity by two extreme scenarios:

- Scenario 1: the neighbor sensors of sensor v_i in $\mathcal{N}_i$ are all connected, i.e., $\forall v_j \in \mathcal{N}_i, \forall v_k \in \mathcal{N}_i \setminus \{v_j\}, v_k \in \mathcal{N}_j$, where $\mathcal{N}_j$ is the set of neighbor sensors of v_j.

- Scenario 2: the neighbor sensors of sensor v_i in $\mathcal{N}_i$ are all isolated to each other, i.e., $\forall v_j \in \mathcal{N}_i, \forall v_k \in \mathcal{N}_i \setminus \{v_j\}, \{v_k\} \nsubseteq \mathcal{N}_j$.

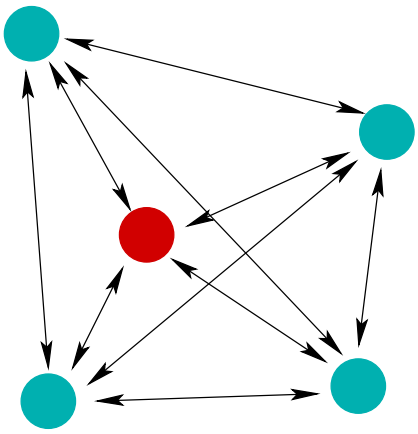

Figure 4.2. Scenario 1: complete graph

4.2.2.2.1 Scenario 1 The first scenario corresponds to a complete graph, as shown in Figure 4.2 constructed by sensor v_i and its neighbor sensors in $\mathcal{N}_i$. In this scenario, all sensors are direct neighbors of all other sensors.

When applying the first sensor selection discussed previously without the involvement of I-Headers, sensor v_i chooses its neighbor for message communication randomly, every neighbor sensor has a probability $\frac{1}{N_i}$ of receiving a message from v_i. When all sensors are connected, any neighbor sensor v_j having the data of sensor v_i indicates that all sensors in $\mathcal{N}_i$ may aggregate the data. Sensor v_j may have received the data from either sensor v_i or another neighbor sensor of v_i. Therefore, in humble wireless sensor networks, the network has *stochastic connectivity* based on the communication range model we discussed in Chapter 2.

In order to analyze the stochastic connectivity, we consider the case in Scenario 1 that sensors in $\mathcal{N}_i$ can only be reached by other sensors in $\mathcal{N}_i$ as well as sensor v_i in the wireless sensor networks. We start with the analysis of the probability of how many sensors within $\mathcal{N}_i \cup \{v_i\}$ has aggregated data after every communication. Because the humble sensors will exchange messages with neighbor sensors, i.e., the sensor transmits and receives from one of its neighbor sensors, the analysis is based on the link between every two sensors. In a complete graph constructed by sensor v_i and its $\mathcal{N}_i$ neighbor sensors, the total number of links is

$$C_i = \frac{(N_i + 1)N_i}{2} \,. \tag{4.5}$$

Therefore, the possibility that two specific sensors among $\mathcal{N}_i \cup \{v_i\}$ exchange their messages is $\frac{1}{C_i}$. Using the example shown in Figure 4.2, we analyze the probability of the number of sensors which have aggregated s_i. We use the term "slot-i" indicating

the i-th time that a sensor among $\mathcal{N}_i \cup \{v_i\}$ wakes up and let $p_i(k)$ indicates that at the end of slot-i, there are k sensors has aggregated data s_i.

- Slot-1: there are at most 2 sensors that have aggregated s_i.

  The possibility that two sensors have v_i is

$$p_1(2) = \frac{N_i}{C_i}, \tag{4.6}$$

  and the probability that only one sensor has s_i aggregated is

$$p_1(1) = \frac{C_i - N_i}{C_i}. \tag{4.7}$$

- Slot-2: at most 3 sensors can have aggregated s_i. We have

$$\begin{aligned} p_2(3) &= p_1(2)\frac{2(N_i-1)}{C_i}, && (4.8)\\ p_2(2) &= p_1(2)\frac{C_i - 2(N_i-1)}{C_i} + p_1(1)\frac{N_i}{C_i}, && (4.9)\\ p_2(1) &= p_1(1)\frac{C_i - N_i}{C_i}. && (4.10) \end{aligned}$$

- Slot-3: s_i is available in 4 sensors. Their probabilities are

$$\begin{aligned} p_3(4) &= p_2(3)\frac{3(N_i-2)}{C_i} && (4.11)\\ p_3(3) &= p_2(3)\frac{C_i - 3(N_i-2)}{C_i} + p_2(2)\frac{2(N_i-1)}{C_i} && (4.12)\\ p_3(2) &= p_2(2)\frac{C_i - 2(N_i-1)}{C_i} + p_2(1)\frac{N_i}{C_i} && (4.13)\\ p_3(1) &= p_2(1)\frac{C_i - N_i}{C_i} && (4.14) \end{aligned}$$

- Slot-4: s_i can be aggregated by 5 sensors, i.e., all sensors in $\mathcal{N}_i \cup \{v_i\}$. The probabilities are

$$\begin{aligned} p_4(5) &= p_3(4)\frac{4(N_i-3)}{C_i} && (4.15)\\ p_4(4) &= p_3(4)\frac{C_i - 4(N_i-3)}{C_i} + p_3(3)\frac{3(N_i-2)}{C_i} && (4.16)\\ p_4(3) &= p_3(3)\frac{C_i - 3(N_i-2)}{C_i} + p_3(2)\frac{2(N_i-1)}{C_i} && (4.17)\\ p_4(2) &= p_3(2)\frac{C_i - 2(N_i-1)}{C_i} + p_3(1)\frac{N_i}{C_i} && (4.18)\\ p_4(1) &= p_3(1)\frac{C_i - N_i}{C_i} && (4.19) \end{aligned}$$

- Slot-5: starting from slot-5, the maximum number of sensors within $\mathcal{N}_i \cup \{v_i\}$ will be constantly 5. The probabilities are

$$p_5(5) = p_4(5) + p_4(4)\frac{4(N_i-3)}{C_i} \quad (4.20)$$

$$p_5(4) = p_4(4)\frac{C_i-4(N_i-3)}{C_i} + p_4(3)\frac{3(N_i-2)}{C_i} \quad (4.21)$$

$$p_5(3) = p_4(3)\frac{C_i-3(N_i-2)}{C_i} + p_4(2)\frac{2(N_i-1)}{C_i} \quad (4.22)$$

$$p_5(2) = p_4(2)\frac{C_i-2(N_i-1)}{C_i} + p_4(1)\frac{N_i}{C_i} \quad (4.23)$$

$$p_5(1) = p_4(1)\frac{C_i-N_i}{C_i} \quad (4.24)$$

From the example with v_i and its $N_i = 4$ neighbor sensors, a set of probabilities can be derived for a general case where sensor v_i has N_i neighbor sensors, which are collected in $\mathcal{N}_i$, and all sensors in $\mathcal{N}_i \cup \{v_i\}$ form a complete graph, i.e., $\forall v_j \in \mathcal{N}_i \cup \{v_i\}, v_j \cup \mathcal{N}_j = \mathcal{N}_i \cup \{v_i\}$. Let $p_l(k)$ denote the probability that after the l-th communication round among sensors in $\mathcal{N}_i \cup \{v_i\}$, k sensors know data s_i. For $l = 1$, at maximum two sensors will have aggregated data s_i with probabilities being

$$p_1(2) = \frac{N_i}{C_i} \quad (4.25)$$

$$p_1(1) = \frac{C_i-N_i}{C_i}\,. \quad (4.26)$$

For $2 \leq l \leq N_i$, at maximum $l+1$ sensors will have aggregated s_i where the probabilities are

$$p_l(l+1) = p_{l-1}(l)\frac{l(N_i-(l-1))}{C_i} \quad (4.27)$$

$$p_l(k) = p_{l-1}(k)\frac{C_i-k(N_i-(k-1))}{C_i} + p_{l-1}(k-1)\frac{(k-1)(N_i-(k-2))}{C_i}, \forall 1 < k < l+1 \quad (4.28)$$

$$p_l(1) = p_{l-1}(1)\frac{C_i-N_i}{C_i}\,. \quad (4.29)$$

For $l > N_i$, the maximum number of sensors which can aggregate s_i is limited to N_i+1,

and the probabilities distribution for different numbers are

$$p_l(N_i+1) = p_{l-1}(N_i+1) + p_{l-1}(N_i)\frac{N_i}{C_i} \tag{4.30}$$

$$p_l(k) = p_{l-1}(k)\frac{C_i - k(N_i-(k-1))}{C_i} + p_{l-1}(k-1)\frac{(k-1)(N_i-(k-2))}{C_i}, \forall 1 < k < N_i+1 \tag{4.31}$$

$$p_l(1) = p_{l-1}(1)\frac{C_i - N_i}{C_i}\,. \tag{4.32}$$

In Figure 4.3, the comparison is given for the probability that the number of sensors has aggregated data s_i from the simulation (solid lines) using 10 sensors as $\mathcal{N}_i \cup \{v_i\}$, and that from theoretical results (circle dot marker) from (4.25) to (4.32).

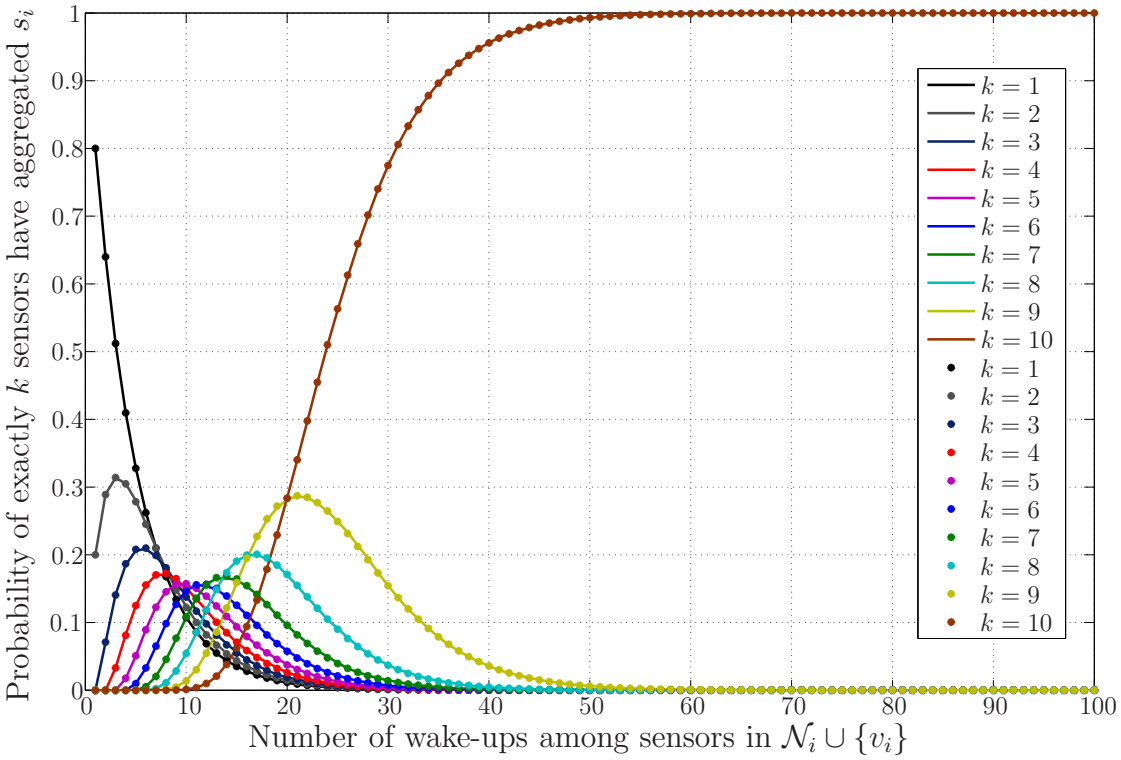

Figure 4.3. Simulation (solid lines) and theoretical (circle dot marker) results of the number of sensors k which has aggregated data s_i for humble sensors without the assistance of I-Header, $N_i + 1 = 10$.

Observing the probabilities from (4.25) to (4.32), it can be concluded that when the connectivity of the humble sensor v_i and its humble neighbor sensors $\mathcal{N}_i$ follows a

complete graph, the probability that after l-th wake-ups of sensors in $\mathcal{N}_i \cup \{v_i\}$ there are $k, k \leq N_i+1$ sensors aggregating the data s_i can be modeled using a Markov chain. At l-th wake-up, the probability $p_l(k)$ consists of two parts. The first part is resulting from that at $l-1$-th wake-up, there is already k sensors aggregated s_i, and at the l-th wake-up there is no new sensor aggregating s_i. The second part is from that at $l-1$-th wake-up, there is $k-1$ sensors aggregated s_i, and at the l-th wake-up, there is one new sensor aggregating s_i. The Markov Chain and the corresponding Transition Matrices can be formulated as follows,

- the probability vector p_1 is

$$p_1 = \begin{bmatrix} p_1(1) \\ p_1(2) \end{bmatrix} \tag{4.33}$$

for $2 \leq l \leq N_i$, the probability vector p_l is of length $l+1$ and can be written as

$$p_l = \begin{bmatrix} p_l(1) \\ p_l(2) \\ \vdots \\ p_l(l+1) \end{bmatrix} \tag{4.34}$$

for $l > N_i$, the probability vector p_l is of fixed length N_i+1 and

$$p_l = \begin{bmatrix} p_l(1) \\ p_l(2) \\ \vdots \\ p_l(N_i+1) \end{bmatrix} \tag{4.35}$$

- the transition matrix for the l-th wake up where $1 \leq l \leq N_i$ is of size $(l+1)$-by-l and is written as

$$T_l^{\mathrm{m}} = \begin{bmatrix} \frac{C_i-N_i}{C_i} & 0 & 0 & \cdots & 0 \\ \frac{N_i}{C_i} & \frac{C_i-2(N_i-1)}{C_i} & 0 & \cdots & 0 \\ 0 & \frac{2(N_i-1)}{C_i} & \frac{C_i-3(N_i-2)}{C_i} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{l(N_i-(l-1))}{C_i} \end{bmatrix} \tag{4.36}$$

the transition matrix for $l > N_i$ is a square matrix of the constant size (N_i+1), given as

$$T_l^{\mathrm{m}} = \begin{bmatrix} \frac{C_i-N_i}{C_i} & 0 & 0 & \cdots & 0 & 0 \\ \frac{N_i}{C_i} & \frac{C_i-2(N_i-1)}{C_i} & 0 & \cdots & 0 & 0 \\ 0 & \frac{2(N_i-1)}{C_i} & \frac{C_i-3(N_i-2)}{C_i} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{N_i(N_i-(N_i-1))}{C_i} & 1 \end{bmatrix} \tag{4.37}$$

From the Markov Chain, one could deduce the probability of p_l without the input of p_{l-1}. Let $a_1 = \frac{N_i}{C_i}$ and $a_{-1} = \frac{C_i - N_i}{C_i}$ be true, inductively $a_l = \frac{l(N_i-(l-1))}{C_i}$ and $a_l = \frac{C_i - l(N_i-(l-1))}{C_i}$ will be true. In order to derive p_l for the l-th wake-up, the number of accumulated items in the l-th row need to be analyzed at first. Table 4.1 gives the number of elements to be summed with the l-th wake-up listed in each row of the transition matrix. In the r-th row, the first 1 appears when $l = r - 1$. The number $t(r, l)$ in the r-th row at $l = k$ is $t(r, l) = t(r, l-1) + t(r-1, l-1)$, where $l \geq r$.

row number	l = 1	l = 2	l = 3	l = 4	l = 5	$\cdots$
1	1	1	1	1	1	$\cdots$
2	1	2	3	4	5	$\cdots$
3	0	1	3	6	10	$\cdots$
4	0	0	1	4	10	$\cdots$
5	0	0	0	1	5	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Table 4.1. Number of elements in summation of each row

Interestingly, Table 4.1 gives a Pascal's Triangle whose elements can be determined by the binomial expansion. Therefore, the number $t(r, l)$ is given by the combination

$$t(r, l) = \begin{pmatrix} l \\ r-1 \end{pmatrix}, \tag{4.38}$$

where $1 \leq r \leq l + 1$ and $t(r, l) = 0$ for all other cases.

For the l-th wake-ups with $l \leq N_i$, the vector p_l can be written as

$$p_l = \begin{bmatrix} (a_{-1})^l \\ a_1 \left(\sum_{i_1+i_2=l-1, i \geq 0} (a_{-1})^{i_1} (a_{-2})^{i_2} \right) \\ \vdots \\ \left(\prod_{k=1}^{r-1} a_k \right) \left(\sum_{i_1+i_2+\cdots+i_r=l-r+1, i_j \geq 0} (a_{-1})^{i_i} (a_{-2})^{i_2} \cdots (a_{-r})^{i_r} \right) \\ \vdots \\ \prod_{k=1}^{l} a_k \end{bmatrix}, \tag{4.39}$$

where r stands for the r-th row. It can be noticed that p_l is a vector of length $l + 1$ when $l \leq N_i$.

When $l > N_i$, the vector p_l has always length $N_i + 1$ and it is

$$p_l = \begin{bmatrix} (a_{-1})^l \\ a_1 \left(\sum_{i_1+i_2=l-1, i \geq 0} (a_{-1})^{i_1} (a_{-2})^{i_2} \right) \\ \vdots \\ \left(\prod_{k=1}^{r-1} a_k \right) \left(\sum_{i_1+i_2+\cdots+i_r=l-r+1, i_j \geq 0} (a_{-1})^{i_i} (a_{-2})^{i_2} \cdots (a_{-r})^{i_r} \right) \\ \vdots \\ \left(\prod_{k=1}^{N_i-1} a_k \right) \left(\sum_{i_1+i_2+\cdots+i_{N_i}=l-N_i+1, i_j \geq 0} (a_{-1})^{i_i} (a_{-2})^{i_2} \cdots (a_{-N_i})^{i_{N_i}} \right) \\ \prod_{k=1}^{N_i} a_k \left(1 + \sum_{m=1}^{l-N_i} \sum_{i_1+i_2+\cdots+i_{N_i}=m, i_j \geq 0} (a_{-1})^{i_i} (a_{-2})^{i_2} \cdots (a_{-N_i})^{i_{N_i}} \right) \end{bmatrix} . \quad (4.40)$$

When the number of wake-ups is $l \to \infty$, the (N_i+1)-th element of p_l is approximating 1.

Since I-Header is not exchanged and used between sensors, every wake-up requires two message transmissions. If I-Headers are exchanged, sensors can acknowledge their aggregated measurement data with I-Headers.

To focus the behavior of the data from sensor v_i, we assume that among sensors in $\mathcal{N}_i \cup \{v_i\}$, all other bits in their I-Header are the same except the bit for data s_i. Initially, sensor v_i has in its I-Header the bit for s_i being 1 and sensors in $\mathcal{N}_i$ have in their I-Headers the bits for s_i being 0. When a sensor initializes a message communication with its neighbors, I-Header helps to check whether new data is contained for each other, as we have introduced in our neighbor sensor selection. After transmitting I-Header from the sensor which initializes the communication to the neighbor sensors, an acknowledgment is given if the communication of one direction is needed. Therefore, the following three cases will happen.

- When both sides have aggregated data s_i, their I-Header will be identical under our assumption. Therefore, no message transmission takes places.
- When one side has aggregated data s_i, the sensor which has aggregated data s_i will transmit its message to the one which has not. Therefore, only one message transmission takes places.
- When neither side has aggregated data s_i, under our assumption, their I-Headers are also identical, so neither of two sensors will transmit messages.

Based on the analysis above, one can conclude that the probability that after l message communications, the number of sensors k which have aggregated data s_i among sensors

in $\mathcal{N}_i \cup \{v_i\}$ will be

$$p_l(k) = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{if } k \neq l \end{cases} . \tag{4.41}$$

In Figure 4.4, we show the probability that all sensors in $\mathcal{N}_i \cup \{v_i\}$ have aggregated data s_i versus the number of message communications that are performed. As shown in the figure, the involvement of I-Header when sensors exchange their messages can significantly reduce the number of message communications that are performed.

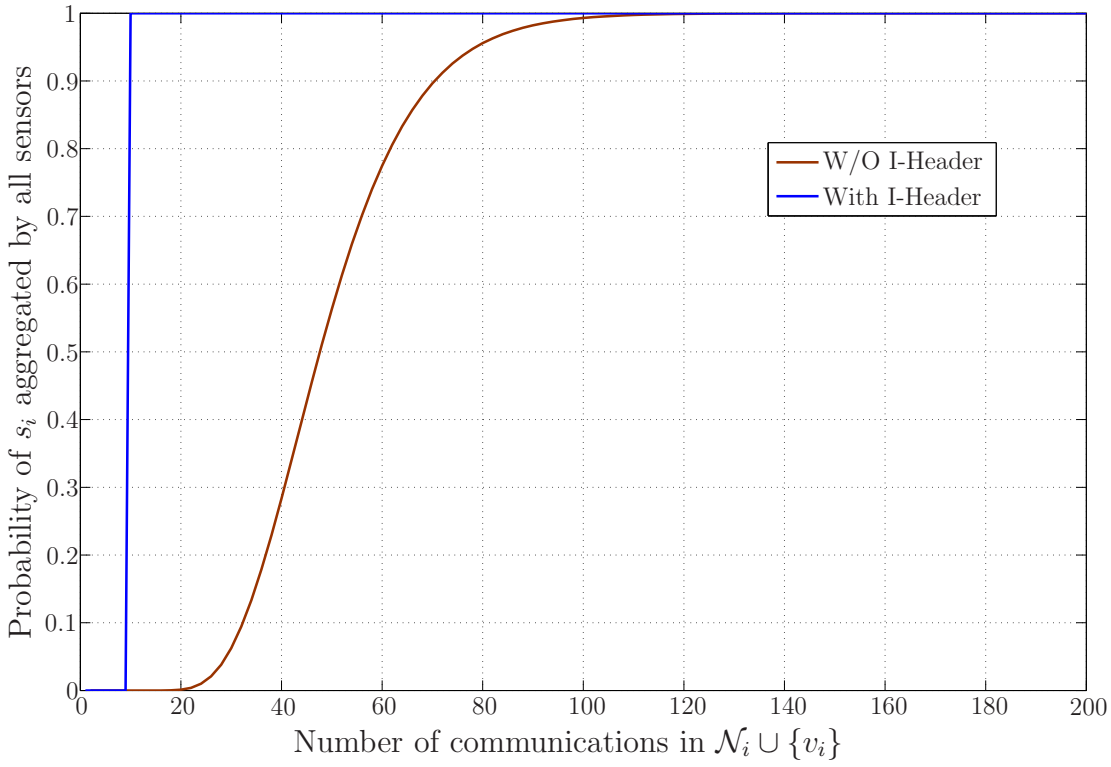

Figure 4.4. Probability of the number of communications required until all sensors in $\mathcal{N}_i \cup \{v_i\}$ have aggregated data s_i, $N_i + 1 = 10$.

With the help of Figure 4.3 and 4.4, we demonstrate the connectivity of humble wireless sensor networks by counting the number of wake-ups and the number of communications that are required for all sensors in $\mathcal{N}_i \cup \{v_i\}$ to aggregate data s_i from sensor v_i in the Scenario 1 exemplified in Figure 4.2.

4.2.2.2.2 Scenario 2 In Scenario 2, neighbor sensors of sensor v_i can only exchange messages with each other via sensor v_i as they occupy no direct connections between

each other. A star graph of sensors among $\mathcal{N}_i \cup \{v_i\}$ centered at sensor v_i results. An example is shown in Figure 4.5.

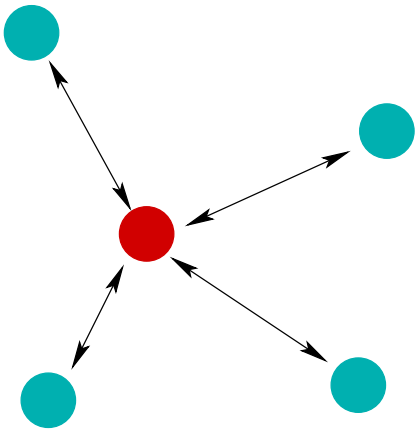

Figure 4.5. Scenario 2: star graph

In this scenario, the total number of connections among sensors $\mathcal{N}_i \cup \{v_i\}$ is

$$C_i = N_i. \tag{4.42}$$

The connection between two sensors is enabled based on the probability that a neighbor sensor of v_i is selected. In the following, we apply similar ideas as in Scenario 1 to analyze the connectivity when no I-Header is involved.

- Slot-1: the probability that two sensors can aggregate s_i is

$$p_1(2) = 1\ , \tag{4.43}$$

  based on the fact that among N_i connections $\mathcal{N}_i \cup \{v_i\}$, there will always be one connection activated.

- Slot-2: there will be at maximum 3 sensors which can aggregate data s_i. The probabilities are

$$p_2(3) = \frac{N_i - 1}{N_i}, \tag{4.44}$$

$$p_2(2) = \frac{1}{N_i}\ . \tag{4.45}$$

- Slot-3: at maximum 4 sensor can aggregate s_i. The probabilities are

$$p_3(4) = p_2(3)\frac{N_i - 2}{N_i}, \tag{4.46}$$

$$p_3(3) = p_2(3)\frac{2}{N_i} + p_2(2)\frac{N_i - 1}{N_i}, \tag{4.47}$$

$$p_3(2) = p_2(2)\frac{1}{N_i}. \tag{4.48}$$

- Slot-4: data s_i can be aggregated at all 5 sensors. The probabilities are

$$\begin{aligned} p_4(5) &= p_3(4)\frac{N_i-3}{N_i}, && (4.49)\\ p_4(4) &= p_3(4)\frac{3}{N_i}+p_3(3)\frac{N_i-2}{N_i}, && (4.50)\\ p_4(3) &= p_3(3)\frac{2}{N_i}+p_3(2)\frac{N_i-1}{N_i}, && (4.51)\\ p_4(2) &= p_3(2)\frac{1}{N_i}. && (4.52) \end{aligned}$$

- Slot-5: data s_i can be aggregated at all 5 sensors among $\mathcal{N}_i \cup \{v_i\}$ with probabilities being

$$\begin{aligned} p_5(5) &= p_4(5)+p_4(4)\frac{N_i-3}{N_i}, && (4.53)\\ p_5(4) &= p_4(4)\frac{3}{N_i}+p_4(3)\frac{N_i-2}{N_i}, && (4.54)\\ p_5(3) &= p_4(3)\frac{2}{N_i}+p_4(2)\frac{N_i-1}{N_i}, && (4.55)\\ p_5(2) &= p_4(2)\frac{1}{N_i}. && (4.56) \end{aligned}$$

From the example of sensor v_i and its $N_i = 4$ neighbor sensors constructing a star neighbor topology, we can formulate the probabilities in general of sensor v_i and N_i neighbor sensors with a star topology. For the first link activation, i.e., $l = 1$, the probabilities are

$$\begin{aligned} p_1(2) &= 1 && (4.57)\\ p_1(1) &= 0\,. && (4.58) \end{aligned}$$

For the l-th link activation, where $2 \le l \le N_i$, among sensor v_i and its neighbors in $\mathcal{N}_i$ there will be at maximum $l+1$ sensors having aggregated data s_i. These $l+1$ sensors are sensor v_i and another l sensors in $\mathcal{N}_i$. The probabilities are

$$\begin{aligned} p_l(l+1) &= p_{l-1}(l)\frac{N_i-(l-1)}{N_i}, && (4.59)\\ p_l(k) &= p_{l-1}(k)\frac{k-1}{N_i}+p_{l-1}(k-2)\frac{N_i-(k-2)}{N_i}, \forall 2<k<l+1 && (4.60)\\ p_l(2) &= p_{l-1}(2)\frac{1}{N_i}. && (4.61) \end{aligned}$$

For $l > N_i$, the probabilities are

$$p_l(N_i+1) = p_{l-1}(N_i+1) + p_{l_1}(N_i)\frac{1}{N_i} \quad (4.62)$$

$$p_l(k) = p_{l-1}(k)\frac{k-1}{N_i} + p_{l-1}(k-2)\frac{N_i-(k-2)}{N_i}, \forall 2 < k < l+1 \quad (4.63)$$

$$p_l(2) = p_{l-1}(2)\frac{1}{N_i}. \quad (4.64)$$

Figure 4.6 gives the probability that the measurement data s_i is aggregated at k sensors after a given number of total wake-ups. The comparison is also given between the results from the simulation with $N_i = 10$ and the results from (4.57) to (4.64).

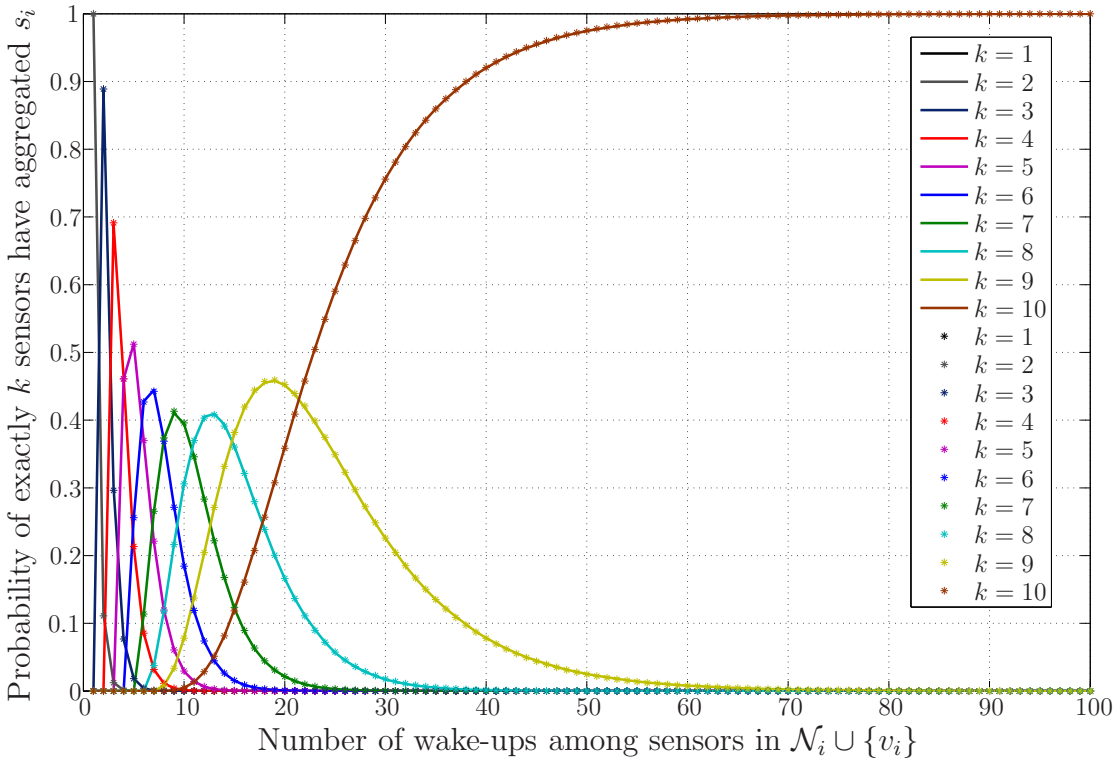

Figure 4.6. Simulation (solid lines) and theoretical (circle dot marker) results of the number of sensors k which has aggregated data s_i for humble sensors without the assistance of I-Header, star topology, $N_i + 1 = 10$.

Since the probability of the l-th link activation depends only on $(l-1)$-th link activation, it can also be formulated using Markov Chain. The probability of the link activation and the transition matrix are given as follows:

- the probability vector p_1 contains only $p_1(2) = 1$ and the vector p_l where $1 < l \leq N_i$ contains l elements which are

$$p_l = \begin{bmatrix} p_l(2) \\ p_l(3) \\ \vdots \\ p_l(l+1) \end{bmatrix} . \tag{4.65}$$

  When $l > N_i$ is true, p_l constantly contains N_i elements starting with $p_l(2)$. It can be written as

$$p_l = \begin{bmatrix} p_l(2) \\ p_l(3) \\ \vdots \\ p_l(N_i+1) \end{bmatrix} . \tag{4.66}$$

- Let $a_2 = \frac{N_i-1}{N_i}$ and $a_{-2} = \frac{1}{N_i}$ and let $a_l = \frac{N_i-(l-1)}{N_i}$ and $a_{-l} = \frac{l-1}{N_i}$. The transition matrix from p_1 to p_2 is

$$T_2^{\mathrm{m}} = \begin{bmatrix} a_{-2} \\ a_2 \end{bmatrix} . \tag{4.67}$$

  The transition matrix from p_{l-1} to p_l with $2 \leq l \leq N_i$ is

$$T_l^{\mathrm{m}} = \begin{bmatrix} a_{-2} & 0 & 0 & \cdots & 0 \\ a_2 & a_{-3} & 0 & \cdots & 0 \\ 0 & a_3 & a_{-4} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_l \end{bmatrix} . \tag{4.68}$$

  For $l > N_i$, the transition matrix is a constant matrix with size N_i-by-N_i, which is

$$T_l^{\mathrm{m}} = \begin{bmatrix} a_{-2} & 0 & 0 & \cdots & 0 & 0 \\ a_2 & a_{-3} & 0 & \cdots & 0 & 0 \\ 0 & a_3 & a_{-4} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & a_{N_i} & 1 \end{bmatrix} . \tag{4.69}$$

The vector p_l without the dependence on p_{l-1} is the multiplication of the transition matrices from T_2^{m} to T_l^{m}. Therefore, for $2 \leq l \leq N_i$ we have

$$p_l = \begin{bmatrix} (a_{-2})^l \\ a_2 \left(\sum_{i_2+i_3=l-1, i_j>0} (a_{-2})^{i_2} (a_{-3})^{i_3} \right) \\ \vdots \\ \left(\prod_{k=2}^{r} a_k \right) \left(\sum_{i_2+i_3+\cdots+i_r=l-r, i_j>0} (a_{-2})^{i_2} (a_{-3})^{i_3} \cdots (a_{-r})^{i_r} \right) \\ \vdots \\ \prod_{k=2}^{l} a_k \end{bmatrix} . \tag{4.70}$$

For $l > N_i$, the vector p_l will be

$$p_l = \begin{bmatrix} (a_{-2})^l \\ a_2 \left(\sum_{i_2+i_3=l-1, i_j>0} (a_{-2})^{i_1} (a_{-3})^{i_2} \right) \\ \vdots \\ \left(\prod_{k=2}^{r} a_k \right) \left(\sum_{i_2+i_3+\cdots+i_r=l-r, i_j>0} (a_{-2})^{i_2} (a_{-3})^{i_3} \cdots (a_{-r})^{i_r} \right) \\ \vdots \\ \prod_{k=2}^{N_i} a_k \left(1 + \sum_{m=1}^{l-N_i} \sum_{i_2+i_3+\cdots+i_{N_i}=m, i_j>0} (a_{-2})^{i_2} (a_{-3})^{i_3} \cdots (a_{-r})^{i_{N_i}} \right) \end{bmatrix} . \quad (4.71)$$

Figure 4.7 compares the connectivity of Scenario-1, Scenario-2 without I-Header and with I-Headers. With I-Headers, the number of communications needed to ensure that all sensors among $\mathcal{N}_i \cup \{v_i\}$ have received s_i is a deterministic value. Without I-Header, redundant communications are performed. Therefore, the total number of communications needed is a probabilistic as shown in the figure.

Furthermore, using I-Headers ensures that the numbers of communications needed for both scenarios are the same. However, without I-Headers, Scenario 1 and 2 will have different performance as shown in the figure.

In scenario 1, the probability curve increases slower at the beginning comparing that in scenario 2 due to that in a complete graph more redundant communication could be performed without s_i being really communicated. After more communications are performed, the performance becomes better in comparison to scenario 2 since more sensors has aggregated s_i and it is more probable that s_i is communicated to a sensor without s_i aggregated yet in a complete graph. This explains the crossing point in the figure.

As mentioned previously, the connectivity in random gossiping can be understood as the availability of data s_i at sensors other than v_i with finite communication rounds. We can conclude that introducing I-Headers gives better connectivity in humble wireless sensor networks.

4.2.2.3 Convergence

The connectivity cannot directly reflect the convergence generally in wireless sensor networks. When random gossiping is applied, there is no centralized architecture for organizing the communications of sensors. We decompose the convergence analysis

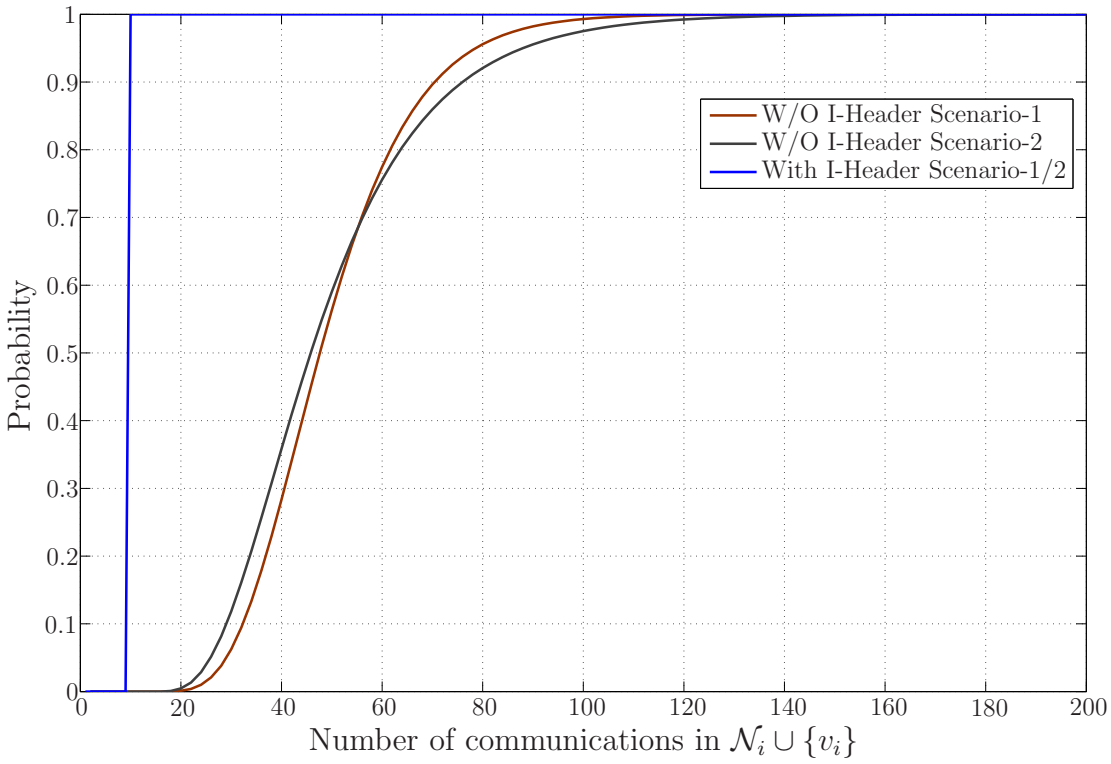

Figure 4.7. Comparison of the probabilities of the numbers of communications required until all sensors in $\mathcal{N}_i \cup \{v_i\}$ have aggregated data s_i, $N_i + 1 = 10$, for Scenario-1 and Scenario-2.

into convergence analyses of measurement data $s_i, i = 1, 2, ...N$ being aggregated at all sensors in the network, we call it *the convergence of data* s_i.

When measurement data s_i is exchanged as a message among sensors, the data s_i will be aggregated by more and more sensors in the network. If we define a function whose input is the time and the output is the portion of sensors that have aggregated data s_i in the network, the function is a monotonically increasing function. Therefore, exchanging a message which has aggregated data s_i among sensors will always improve the convergence in a way that there are new sensors receiving a message containing the data s_i every time communication takes places.

There are two possible scenarios when communication is initialized in humble wireless sensor networks. These two scenarios can be depicted as the example shown in Figure 4.8, where sensors v_j, v_{j3}, and v_{j4} are the ones which have already aggregated data s_i and sensors v_{j1} and v_{j2} are the sensors which have not.

When I-Header is not involved in the random gossiping, a sensor, e.g., sensor v_j, may

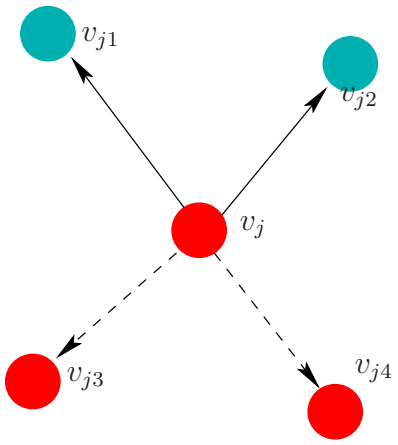

Figure 4.8. Two possible scenarios of a humble initiates communication. Red sensors have aggregated data s_i, and green sensors have not.

choose either a red sensor or a green sensor to communicate. When sensor v_{j3} or v_{j4} is chosen, this message communication is not beneficial to the convergence of the measurement data s_i. We assuming there are in total N_j^{r} red sensors which have already aggregated the data s_i and N_j^{g} green sensors which have not yet aggregated s_i. Apparently, $N_j^{\mathrm{r}} + N_j^{\mathrm{g}} = N_j$ holds, and the probability that there will be a beneficial communication of data s_i performed by sensor v_j is

$$p_{j,s_i}^{\mathrm{b}} = \frac{N_j^{\mathrm{g}}}{N_j} \,. \tag{4.72}$$

In a humble wireless sensor network, a useful message communication of s_i will lead to a new sensor having s_i aggregated. If one counts the number of beneficial communications of s_i and the number of sensors that have already aggregated data s_i, their difference is always 1. We see the convergence of data s_i as a procedure that the sensor which has s_i and the sensor which does not have s_i interacts with each other. Let the set of sensors which have s_i at a specific time be denoted by $\mathcal{V}_{s_i}$ with the number being V_{s_i}. The sensors which have not aggregated s_i are $\mathcal{V} \setminus \mathcal{V}_{s_i}$. Let $\partial\mathcal{V}_{s_i}$ denote all connections between sensors in $\mathcal{V}_{s_i}$ and sensors in $\mathcal{V} \setminus \mathcal{V}_{s_i}$. A ratio $\frac{|\partial\mathcal{V}_{s_i}|}{|\mathcal{V}_{s_i}|}$ similar to Cheeger Constant is used to approximate the bottleneck of a beneficial communication of data s_i. As shown in Figure 4.9, sensors v_i, v_j, and v_k have already aggregated s_i, sensors v_l, v_m, and v_n have not yet aggregated s_i. The value $\frac{|\partial\mathcal{V}_{s_i}|}{|\mathcal{V}_{s_i}|}$ is 2, indicating that among all sensors in the Figure, the number of communications which may result in a new sensor having s_i aggregated are twice as many as the ones which may not.

To continue with the analysis of the convergence, we consider a time slot when in the network there are $\mathcal{V}_{\mathrm{s}}$ sensors have aggregated data s_i. The possibility that there is a

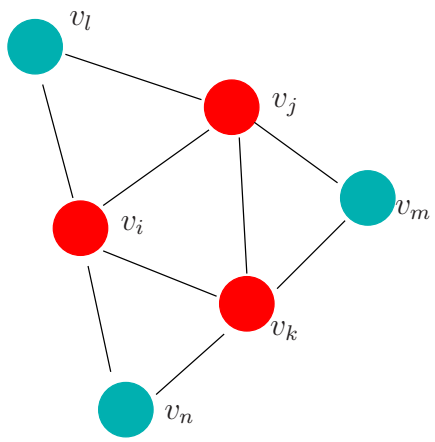

Figure 4.9. Demonstration of $\frac{|\partial \mathcal{V}_{s_i}|}{|\mathcal{V}_{s_i}|}$

new sensor aggregating data s_i is

$$p_{|V_s|+1} = \frac{|\partial \mathcal{V}_{s_i}|}{|E|}, \tag{4.73}$$

where $|E|$ is the total number of connections between all sensors. According to the geometric distribution, on average $\frac{1}{p_{|V_s|+1}}$ communications are required for a new sensor to aggregate data s_i, when there are $|\mathcal{V}_s|$ sensors have already aggregated data s_i. Under the assumption of independent communication initialization when a sensor wakes up randomly, the average number of communications required for all sensors in the network is

$$C_{s_i} = \sum_{l=1}^{N-1} \frac{1}{p_{l+1}}, \tag{4.74}$$

where

$$p_{l+1} = \frac{|\partial \mathcal{V}_{s_i}^l|}{|E|}, \tag{4.75}$$

and $\mathcal{V}_{s_i}^l$ is the set of l sensors which have already aggregated data s_i. To approximate $|\partial \mathcal{V}_{s_i}^l|$, two extreme cases are considered to find a loose boundary of $|\partial \mathcal{V}_{s_i}^l|$.

- A lower bound is given by the minimum $|\partial \mathcal{V}_{s_i}^l|$ when all sensors in $\mathcal{V}_{s_i}^l$ are *maximally* connected to each other.
- A upper bound is given by the maximum $|\partial \mathcal{V}_{s_i}^l|$ when all sensors in $\mathcal{V}_{s_i}^l$ are connected in such a way that they form a path, i.e., except the head and the tail sensor that connect only one other sensor in $\mathcal{V}_{s_i}^l$, every other sensor is connected to another two sensors in $\mathcal{V}_{s_i}^l$.

To facilitate the analysis, we assume also that every sensor in the network has in average N^{av} neighbor sensors, where N^{av} is a variable dependent on the communication range d. It shall be noticed that this assumption does not necessarily lead to a regular graph where every sensor has the same number of neighbor sensors. Under these two additional assumptions, in the first case where all sensors in $\mathcal{V}^l_{s_i}$ are *maximally* connected to each other, when $N^{\text{av}} \geq |\mathcal{V}^l_{s_i}|$, every sensor in $\mathcal{V}^l_{s_i}$ is connected to all other $|\mathcal{V}^l_{s_i}|-1$ sensors in $\mathcal{V}^l_{s_i}$, and each sensor in $\mathcal{V}^l_{s_i}$ connected only to the other $N^{\text{av}}-|\mathcal{V}^l_{s_i}|+1$ sensors outside of $\mathcal{V}^l_{s_i}$. Hence, we have

$$|\partial\mathcal{V}^l_{s_i}| \geq |\mathcal{V}^l_{s_i}| \left(N^{\text{av}} - |\mathcal{V}^l_{s_i}| + 1\right). \tag{4.76}$$

When $N^{\text{av}} < |\mathcal{V}^l_{s_i}|$, it could be possible that some sensors in $\mathcal{V}^l_{s_i}$ connect only to other sensors in $\mathcal{V}^l_{s_i}$, and only a few sensors in $\mathcal{V}^l_{s_i}$ connect to sensors which are not in $\mathcal{V}^l_{s_i}$. The loose bound of $|\partial\mathcal{V}^l_{s_i}|$ is then

$$|\partial\mathcal{V}^l_{s_i}| \geq 1. \tag{4.77}$$

For the second case, sensors in $\mathcal{V}^l_{s_i}$ forms a path. Therefore, except the head and tail sensors of the path which have maximum $N^{\text{av}}-1$ neighbors being not in $\mathcal{V}^l_{s_i}$, other sensors in the path have $N^{\text{av}}-2$ neighbors which are not in $\mathcal{V}^l_{s_i}$. The loose bound is hence given as

$$|\partial\mathcal{V}^l_{s_i}| \leq 2(N^{\text{av}}-1) + (|\mathcal{V}^l_{s_i}|-2)(N^{\text{av}}-2). \tag{4.78}$$

Equations from (4.76) to (4.78) provide a pair of loose bounds of the convergence of data s_i, when all sensors are humble sensors and no I-Header is involved.

When I-Headers are in use, the convergence of s_i would be loosely lower bounded by the number of sensors in the network. Here the bound is *loose* because redundant transmissions of s_i may happen, when in a message there are other measurement data need to be aggregated by the sensor that receives the I-Header.

4.2.3 Random gossiping in greedy wireless sensor networks

In greedy wireless sensor networks, a sensor communicates with all of its neighbor sensors when it initializes the communications. In this subsection, we organize our discussion with the following questions.

- How a greedy sensor communicates with its neighbor sensors?

- What is the connectivity of the network?
- What is the speed of convergence?
- What is the performance of bias cancellation with greedy sensors?

4.2.3.1 Greedy types

We use the name of *greedy* to describe that the sensor which initiates the communications are communicating with all its neighbor sensors. However, due to the one-to-many communication type, the communications between a sensor and its neighbor sensors require scheduling to adjust the timing of transmissions between the neighbor sensors and the sensor which initiates the communications.

For the sensor which initiates the communications, it has two possible roles when it communicates with its neighbor sensors. It can be either

- a *greedy listener*, which receives all messages sent from its neighbor sensors, or
- a *greedy speaker* which only broadcasts its own message to all its neighbor sensors without receiving any messages.

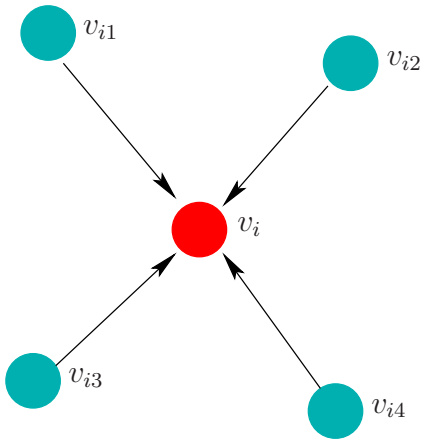

Figure 4.10. Sensor v_i as a greedy listener

When a sensor v_i is purely a greedy listener as shown in Figure 4.10, it triggers its neighbor sensors v_{i1}, v_{i2}, v_{i3} and v_{i4} to send their messages without sending any message back. If we only consider a time division mode of transmissions, the *trigger* that the

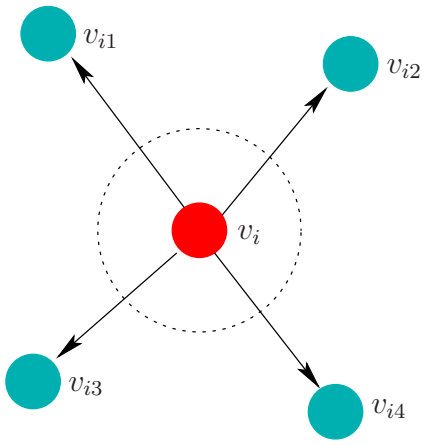

Figure 4.11. Sensor v_i as a greedy speaker

sensor sent to its neighbor sensors should contain a scheduling information telling which time slot a neighbor sensor should use to transmit its message.

For a purely greedy speaker as sensor v_i shown in Figure 4.11, a broadcast is performed by the sensor which initiates the communications, and no feedback message is transmitted by the neighbor sensors v_{i1}, v_{i2}, v_{i3} or v_{i4}. Only one message transmission is needed in this situation.

A combination of the greedy listener and the greedy speaker is considered in this thesis. When a sensor initiates communications with its neighbor sensors, it firstly triggers the message transmissions from its neighbor sensors. After the sensor aggregates all messages it received, it broadcasts the newly aggregated message to all of its neighbor sensors.

4.2.3.2 Connectivity for greedy wireless sensor networks

Intuitively, greedy sensors have better connectivity than humble sensors in terms of all neighbor sensors of sensor v_i aggregating data s_i, due to the greedy many-to-one communication type. It is valid for both greedy speaker case and the case with the combination of greedy speaker and greedy listener. If sensor v_i wakes up as a greedy speaker, it broadcasts its message containing the aggregated data. Therefore, only one broadcast transmission is necessary to aggregate data s_i regardless of whether I-Header is used in the random gossiping. In comparison to the case of all humble sensors, greedy sensors can offer better connectivity in terms of aggregate data s_i at all neighbor sensors of v_i.

4.2.3.3 Convergence

Every time when a greedy sensor initiates communication, the data of the center sensor will be available at all its neighbor sensors. In this case, the approach from the analysis of the humble-sensors case cannot be used. To facilitate the analysis of greedy sensors, all sensors are categorized into three scenarios. The first scenario includes the sensors which have already aggregated data s_i, and all their neighbor sensors have also aggregated data s_i. The second scenario includes sensors which either have already aggregated s_i and they have neighbor sensors which have not aggregated data s_i, or sensors which have not yet aggregated s_i but there are neighbor sensors of them have s_i aggregated already. The third scenario contains the sensors which have not aggregated s_i nor do their neighbor sensors. When sensors in the first scenario wake up and initiate communications with their neighbors, the number of sensors which aggregate s_i will not increase, so as the sensors in the third scenario. On the contrary, when the sensors in the second scenario initiate the communications, there will be sensors in its neighbor sensors, including itself becoming new sensors to have aggregated s_i.

The effect of the wake-up and communications in the network can be seen as the change of the number of sensors in each of the three scenarios. If a sensor v_j in the second scenario wakes up and initiates communications with its N_j neighbor sensors, all N_j+1 sensors will all aggregate data s_i after the communications.

In Figure 4.12 a simple simulation is performed to demonstrate the change of the number of sensors in each scenario using a sensor network with 100 sensors randomly deployed in an area, and the communication range is set to achieve the minimum connectivity. The Figure focuses on the aggregation of data s_i. The horizontal axis gives the number of useful wake-ups with respect to data s_i. A useful wake-up indicates that the center sensor and its neighbor sensors are of the second scenario. The vertical axis gives the number of sensors in each scenario as the number of useful wake-ups increases. The summed number of sensors at each number of useful wake-ups shall be the total number of sensors in the network, i.e., 100 sensors. Such definition helps us to avoid step-like curves in the figure since the wake-up of a sensor in the first scenario and the third scenario will not lead to a change to the number of sensors in each scenario.

As shown in the Figure, the convergence is achieved with certainty when sensors in the network are greedy sensors. In the depicted scenario, the number of sensors of the second scenario is increasing to a maximum value of around 30. It decreases afterwards since sensors fall into the first scenario as the number of useful wake-ups increases.

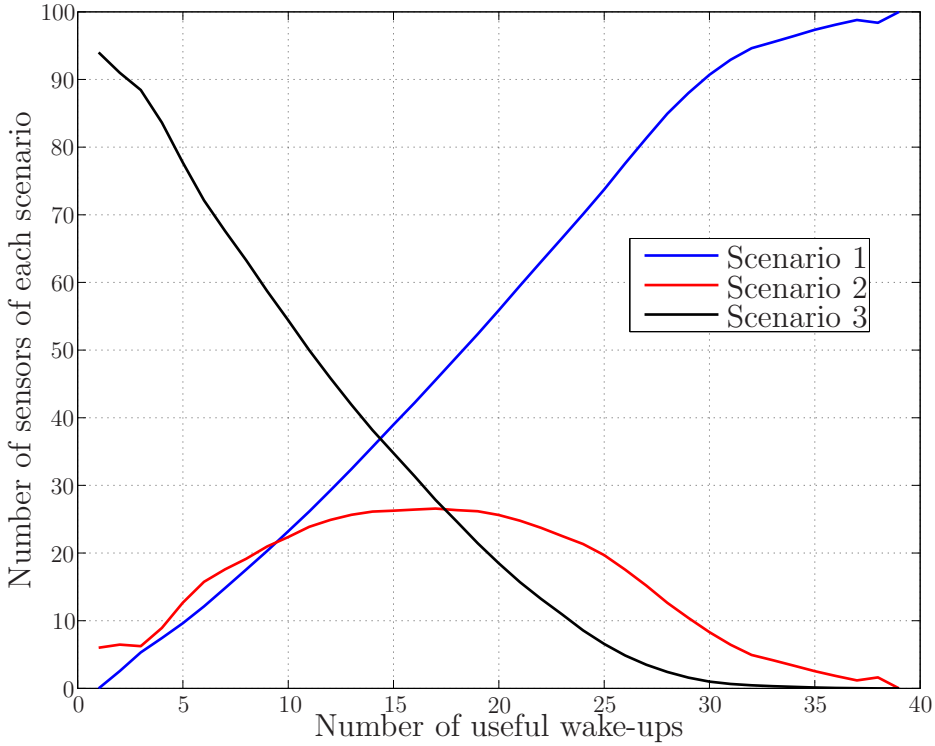

Figure 4.12. Number of sensors of each scenario as aggregation continues

Generally speaking, the higher the largest number of sensors of the second scenario is, the faster the convergence will be. It is because the number of sensors in the second scenario determines in greedy sensor case how fast the sensors in the third scenario will have the data aggregated. One way to achieve this is by increasing connectivity. In Figure 4.13, the communication range is increased in comparison to that of the same sensor network used in Figure 4.12. As a result, the network has larger connectivity. As shown in Figure 4.13, the sensors of the second scenario reach a larger maximum value with a smaller number of useful wake-ups. Additionally, a smaller number of useful wake-ups is needed to achieve the convergence, i.e. all sensors are of the first scenario which have aggregated s_i.

The curves at a larger number of useful wake-ups are not smooth due to the insufficient number of simulations.

When I-Header is used, if a wake-up is not a useful one, i.e. sensors are either of scenario 1 or scenario 3, no message communication will be initiated. Since in Figure 4.12 and Figure 4.13 only useful wake-ups are depicted, the changes in the number of sensors in each scenario will be the same with I-Header applied.

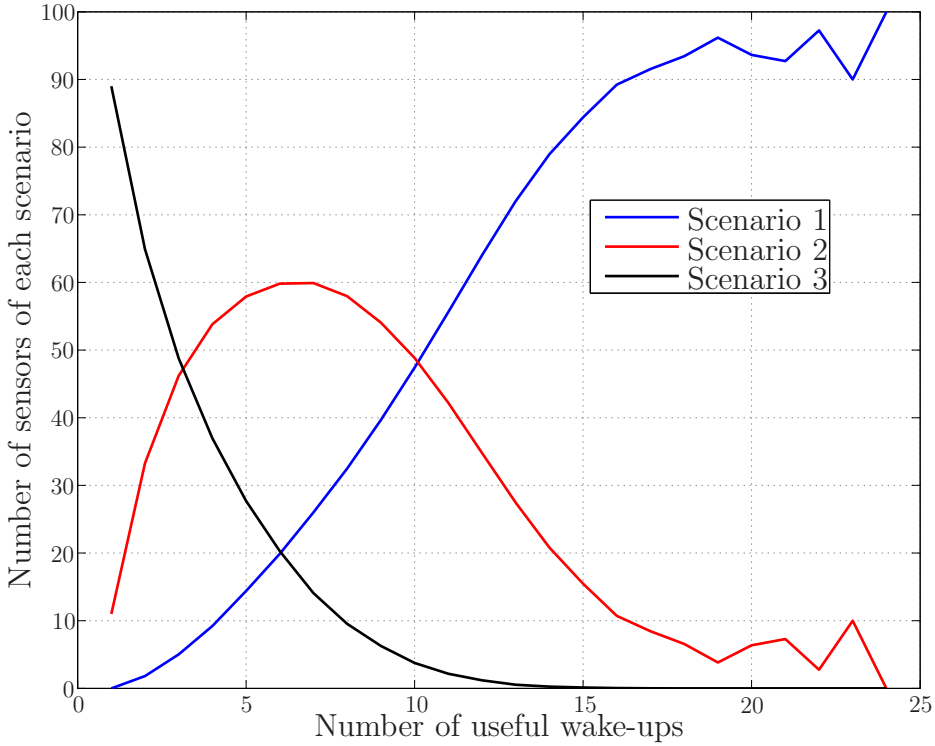

Figure 4.13. The number of sensors of each group as aggregation continues with larger connectivity

4.3 Multihop coordination in random gossiping

As it is discussed in previous sections, random gossiping as a communication paradigm with less topology and stability requirements provides a robust and flexible aggregation method in wireless sensor networks where the aggregation output is finally available to all sensors in the network. So far, the random gossiping is constrained that when a sensor wakes up to initiate the communications, it only interacts with its one-hop neighbor sensors. To reduce the number of communications in the network, in this section, the random gossiping with multihop coordination is proposed to combine the robust random gossiping and the routing-based aggregation method.

By using multihop coordination in random gossiping [CKK13a], communications are extended from a sensor and its one-hop neighbor sensor to the sensors, which are several hops away from the center. The motivation of such multihop coordination is to increase aggregation efficiency. For one sensor in the network, it is most efficient to set this sensor as the root and build a routing tree to connect all sensors in the

network when it is about to aggregate all messages in the network. For a sensor network with N sensors, when a tree is built with sensor v_i being the root, it requires N message communications until sensor v_i aggregates all messages. Because the wake-up is random without centralized scheduling, random gossiping cannot guarantee such a small number of communications for sensor v_i, even with the I-Header we have introduced.

We define the *depth* $\delta_j(i)$ between sensor v_i and sensor v_j as the minimum number of hops with which v_i and v_j are able to communicate. Furthermore, we define the coordination depth c_i of sensor v_i as the maximum depth that sensor v_i can coordinate message exchanges with other sensors when v_i wakes up. The coordination depth c_i can be determined by several realistic parameters, such as the priority of the sensor v_i in the network with respect to the running application, the buffer or memory size, the computation power, et cetera. Given the coordination depth c_i, sensor v_i can exchange messages with all sensors v_j whose depth $\delta_j(i)$ is smaller than c_i when v_i wakes up to initiate the communications. With $\mathcal{N}_i^l$ denoting the set of sensors whose depth with respect to v_i is l, all sensors in $\cup_{l=1}^{c_i}\mathcal{N}_i^l$ are the possible sensors that can exchange messages with v_i when v_i initiates the communications.

In order to coordinate sensors within multiple hops, we use *query* messages to construct the tree rooted as sensor v_i. In general, this query message contains the information such as the root sensor v_i and whether a sensor at certain depth should continue to forward the query message. The latter can be realized by using a decremental counter in the query message.

Since sensors within multiple hops are coordinated, we consider the failure of such coordination to gain a better approximation of the realistic scenario in wireless sensor networks. Due to the failure of communications, and the failure of the wake-ups, the query message may not be received or responded by a sensor, namely, the coordination failure. We model the rate of this coordination failure with a parameter which is referred to as *failure rate* r_i of sensor v_i which indicates the probability that sensor v_i fails to receive query information or to respond to another sensor in the network.

4.3.1 Humble sensor case

In the case of a humble sensor, let $\mathcal{P}_i$ denote the path that is initiated by sensor v_i with the maximum possible depth being c_i, and let $\mathcal{P}_i(0) = v_i$ and $\mathcal{P}_i(l)$ denote the root sensor and the sensor on the l-th hop of this path, respectively. The awake

sensor v_i broadcasts a query message to all its neighbor sensors in $\mathcal{N}_i^1$. In this query message, the information is contained that sensor v_i asks all sensors in $\mathcal{N}_i^1$ for their indicating headers and the information telling sensors in $\mathcal{N}_i^1$ that a path is going to be constructed. Sensors in $\mathcal{N}_i^1$ receive this query information and respond to it with failure rate $r_j, v_j \in \mathcal{N}_i^1$. We denote the set of sensors in $\mathcal{N}_i^1$ who successfully receive and respond to this requirement from v_i by $P(\mathcal{N}_i^1)$. Sensors $v_j \in P(\mathcal{N}_i^1)$ will send back their indicating headers $\boldsymbol{I}_j$. Sensor v_i will choose the sensor which sends back its indicating header and meanwhile results in the greatest bi-directional message differences to be its next hop. The chosen sensor broadcasts the query information to $\mathcal{N}_i^2$ and chooses its next hop. Such process continues until either the maximum coordination depth c_i is reached, or no more sensors respond to join the path when c_i has not been reached. In general, the criterion to choose the sensor for the l-th hop is given by

$$\mathcal{P}_i(l) = \arg \max_{v_j \in P(\mathcal{N}_i^l) \cap \mathcal{N}_{\mathcal{P}_i(l-1)}} \boldsymbol{I}_{\mathcal{P}_i(l-1)} \mathrm{XOR}^b \boldsymbol{I}_j \ , \tag{4.79}$$

where the operation XOR^b performs the XOR-operation to the bit-sequence in $\boldsymbol{I}_i$ and $\boldsymbol{I}_j$ and gives the number of positive bits in the output. The algorithm for constructing path $\mathcal{P}_i$ is given in Algorithm 8.

Algorithm 8 Algorithm of constructing a path initiated by sensor v_i

1: $\mathcal{P}_i(0) = v_i$;
2: $l = 1$;
3: **while** $l \leq c_i$ **do**
4: $\quad \mathcal{P}_i(l-1)$ broadcasts query messages to sensors in $\mathcal{N}_i^l$
5: $\quad$ Determine the set $P(\mathcal{N}_i^l)$
6: $\quad$ Determine $\mathcal{P}_i(l)$ with (4.79)
7: $\quad l = l + 1$
8: **end while**

The achieved path depth is denoted by c_i^α, where $c_i^\alpha \leq c_i$. We denote v_i as the header of path $\mathcal{P}_i$ and $\mathcal{P}(c_i^\alpha)$ as the tail sensor, respectively. Once the path $\mathcal{P}_i$ is constructed, the transmission of the messages starts from the tail sensor. $\mathcal{P}(c_i^\alpha)$ transmits its message to $\mathcal{P}(c_i^\alpha - 1)$, sensor $\mathcal{P}(c_i^\alpha - 1)$ aggregates it with its own message, and the combined message is transmitted to $\mathcal{P}(c_i^\alpha - 2)$. This procedure is done until sensor v_i has the aggregated message whose indicating header is

$$\boldsymbol{I}_i = \Theta \left(\cup_{l=0}^{c_i} \Theta^{-1} \left(\boldsymbol{I}_{\mathcal{P}_i(l)} \right) \right) \ . \tag{4.80}$$

Afterwards, sensor v_i starts to transmit its message along the path $\mathcal{P}_i$ towards the tail sensor until every sensor $v_j \in \mathcal{P}_i$ updates the aggregated message using indicating header $\boldsymbol{I}_j = \boldsymbol{I}_i$.

4.3.2 Greedy sensor case

In the case that sensors in the network are greedy sensors, each sensor $v_i \in V$ will attempt to coordinate a tree whose root is v_i with the maximum depth c_i. Let $\mathcal{T}_i$ be the tree rooted at sensor v_i. $\mathcal{T}_i(l)$ denotes the set of sensors whose depth with respect to sensor v_i is l and $\mathcal{T}_i(0) = v_i$. The father sensor of sensor v_j in the tree $\mathcal{T}_i$ is denoted by $\mathcal{T}_i^f(v_j)$. When sensor v_i wakes up, it broadcasts a query message with its own indicating header $\boldsymbol{I}_i$. Sensors in $\mathcal{N}_i^1$ receive the query message and respond to it with failure rate r_j, where $v_j \in \mathcal{N}_i^1$. The set of sensors who successfully receive and respond to the query is denoted by $P(\mathcal{N}_i^1)$ and hence $\mathcal{T}_i(1) = P(\mathcal{N}_i^1)$. Each sensor in $\mathcal{T}_i(1)$ continues this tree construction by forwarding the query message with its indicating header. If a sensor v_j whose depth with respect to v_i is l, i.e., $v_j \in P(\mathcal{N}_i^l), l = 2, \cdots, c_i$, receives query messages and indicating headers from several sensors, it decides itself which sensor shall be its father sensor $\mathcal{T}_i^f(v_j)$. Let $\mathcal{N}_j^{\mathcal{T}_i} = \mathcal{N}_j \cap P(\mathcal{N}_i^{l-1})$ denote the set of sensors from which sensor v_j receives indicating messages. The criterion of sensor v_j to choose its father sensor is given as

$$\mathcal{T}_i^f(v_j) = \arg \max_{v_k \in \mathcal{N}_j^{\mathcal{T}_i}} \boldsymbol{I}_k \mathrm{XOR}^b \boldsymbol{I}_j \ . \tag{4.81}$$

The algorithm of constructing the tree $\mathcal{T}_i$ is given in Figure 9

Algorithm 9 Algorithm of constructing a tree initiated by sensor v_i

```
1: T_i(0) = v_i;
2: l = 1;
3: while l ≤ c_i do
4:    for v_l ∈ T_i(l − 1) do
5:       v_l broadcasts query message to sensors in N_i^{l+1} ∩ N_l
6:       Determine the set P(N_i^{l+1} ∩ N_l)
7:    end for
8:    for v_m ∈ ∪_{v_l ∈ T_i(l−1)} P(N_i^{l+1} ∩ N_l) do
9:       Determine N_m^{T_i}
10:      Determine T_i^f(v_j) with (4.81)
11:   end for
12:   l = l + 1
13: end while
```

The achieved tree depth is denoted by c_i^β, where $c_i^\beta \leq c_i$. The communication in $\mathcal{T}_i$ starts with sensors in $\mathcal{T}_i(c_i^\beta)$ transmitting their messages to their father sensors in $\mathcal{T}_i(c_i^\beta - 1)$. After a sensor in $\mathcal{T}_i(c_i^\beta - 1)$ receives messages from all its children, it forwards the aggregated messages to its father sensors. This procedure ends until

sensor v_i receives messages from all its children. Sensor v_i will have an aggregated message whose indicating header is

$$\boldsymbol{I}_i = \Theta\left(\cup_{v_j \in \boldsymbol{T}_i} \Theta^{-1}\left(\boldsymbol{I}_j\right)\right) \ . \tag{4.82}$$

Afterwards, sensor v_i broadcasts its newly aggregated message which contains data aggregated of all messages from sensors in $\mathcal{T}_i$ to $\mathcal{T}_i(1)$. Sensors in $\mathcal{T}_i(1)$ forward this message to their children sensors. This procedure stops when all sensors in $\mathcal{T}_i$ have received the message from v_i. Every sensor $v_j \in \mathcal{T}_i$ will now have an updated aggregation data with indicating header $\boldsymbol{I}_j = \boldsymbol{I}_i$. In both humble and greedy cases, we assume that the sensors only suffer from failures in the phase of constructing paths or trees.

4.4 Performance and discussion

In this section, we demonstrate the behavior of the random gossiping and the using of the I-Headers which have been discussed in this chapter.

In Figure 4.14, simulations are performed with $N = 30$ sensors randomly deployed in an area, and the communication range is set for minimum connectivity of the network. A comparison is given to the numbers of communications required until the convergence of the network. In the figure, the blue curve is for the greedy sensors with I-Headers used to reduce the number of message communications, and the red curve is for the humble sensors with I-Headers. The performance of the random gossiping discussed in [BGPS06], where sensors perform similar to greedy sensors, is shown in black. The abscissa in Figure 4.14 is the number of message communications performed in the network. The ordinate gives the probability of the network achieving convergence, i.e., all sensors have aggregated the measurement data of the entire network. A significant reduction of message communications can be witnessed by using I-Headers in random gossiping. Moreover, by comparing blue and red curves, a network with greedy sensors requires fewer number of message communications in comparison to that with only humble sensors.

In Chapter 2, the Assumption 6 is made that the size of the I-Headers is significantly smaller than the length of the application messages. In Figure 4.14, the additional communications for sensors to exchange the I-Headers have been neglected under this assumption.

In Figures 4.15 and 4.16, we consider the impact I-Header transmission on the total number of communications to understand the validity of Assumption 6. The symbol

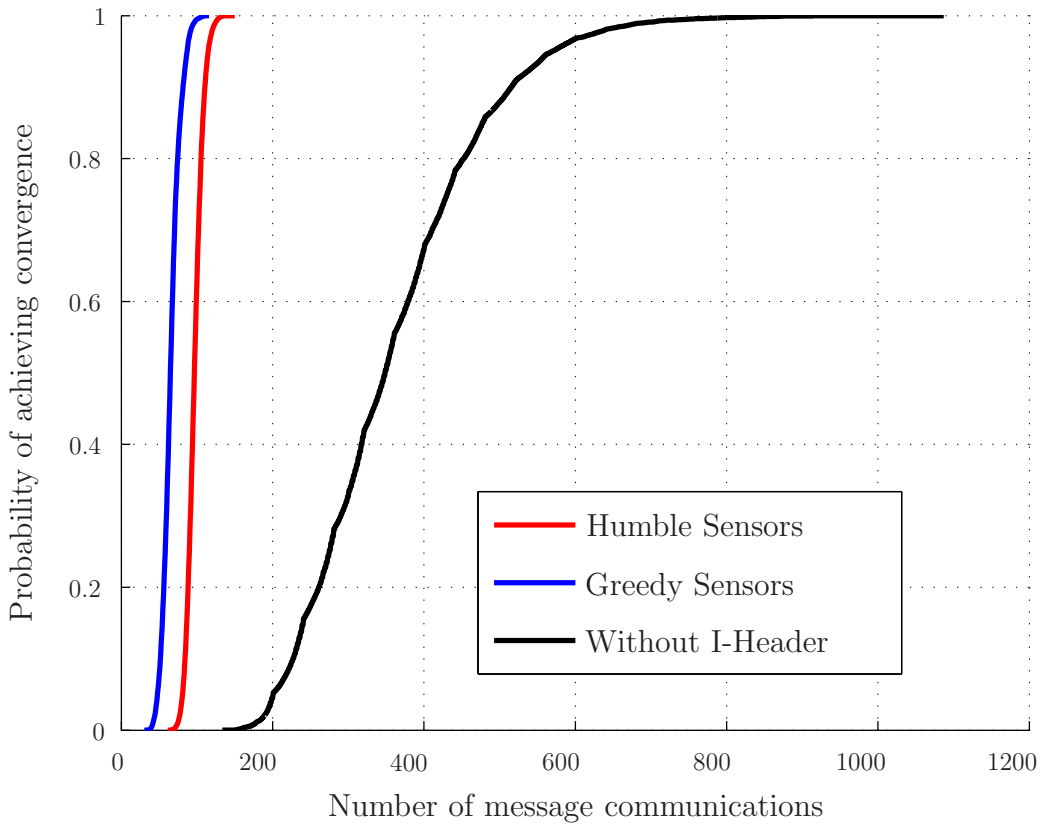

Figure 4.14. Comparison of the numbers of communications required until the gossiping stops

η is used to denote the ratio between the bit length of the I-Header and the length of the messages with the assumption that all aggregations will result in the same message length in bits. By adding the number of communications of the I-Header times η to the number of communications for the application messages, we include the effect of I-Headers into our results.

In Figure 4.15, we demonstrate the impact of I-Headers with different η when sensors are humble. Similar results can be seen in Figure 4.16 when sensors are greedy. Both figures show that the gain in reducing the number of communications when considering the effect of indicating header can still be obtained even with larger η. Furthermore, the greedy sensor strategy is more efficient in terms of aggregation due to its faster spreading of messages within $v_i \cup \mathcal{N}_i$ for every v_i.

Comparing results shown in Figure 4.15 and Figure 4.16, a network with greedy sensors requires fewer communications than a network with only humble sensors for all cases with different η.

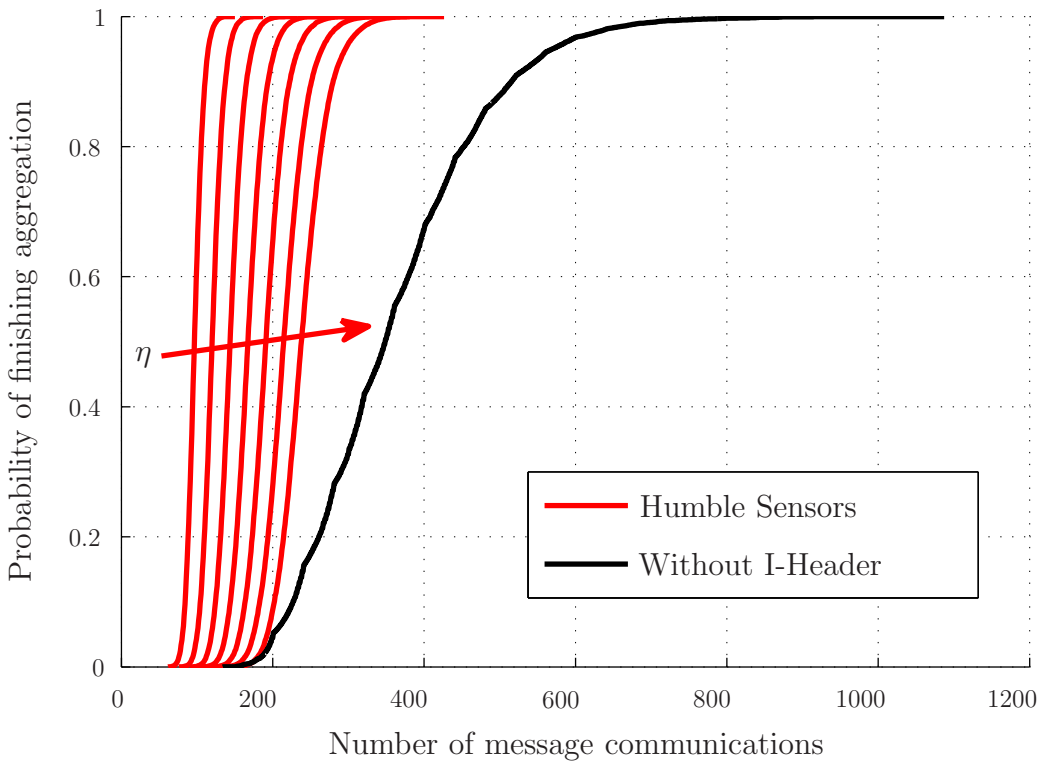

Figure 4.15. Impact of indicating headers with humble sensors. From left to right, $\eta = 0\%, 5\%, 10\%, 15\%, 20\%, 25\%, 30\%$

In the simulation of the multihop coordination in Section 4.3, $N = 30$ sensors are randomly deployed in a two-dimensional squared area in the simulation.

In Figure 4.17 and 4.18, we depict the relationship between the maximum coordination depth of all sensors in the network, i.e., $c_i, v_i \in V$ and the achieved coordination depth in the network under different failure rates r_i of sensors for a network with only humble sensors and greedy sensors, respectively. As shown in the figures, with both humble and greedy sensor strategies, it is unlikely that the maximum coordination depth can be achieved when the failure rate increases. In comparison, the greedy sensor strategy results in a larger achieved coordination depth since all neighbor sensors who decide not to reject the requirement will join the tree.

Figure 4.19 demonstrates the number of message communications in the network to perform until the completeness of aggregation when all sensors are humble sensors, i.e., a path is constructed when a sensor wakes up. As shown in the figure, when the failure rate increases, more message communications are needed due to the failure of communication. Meanwhile, increasing the maximum coordination depth decreases the

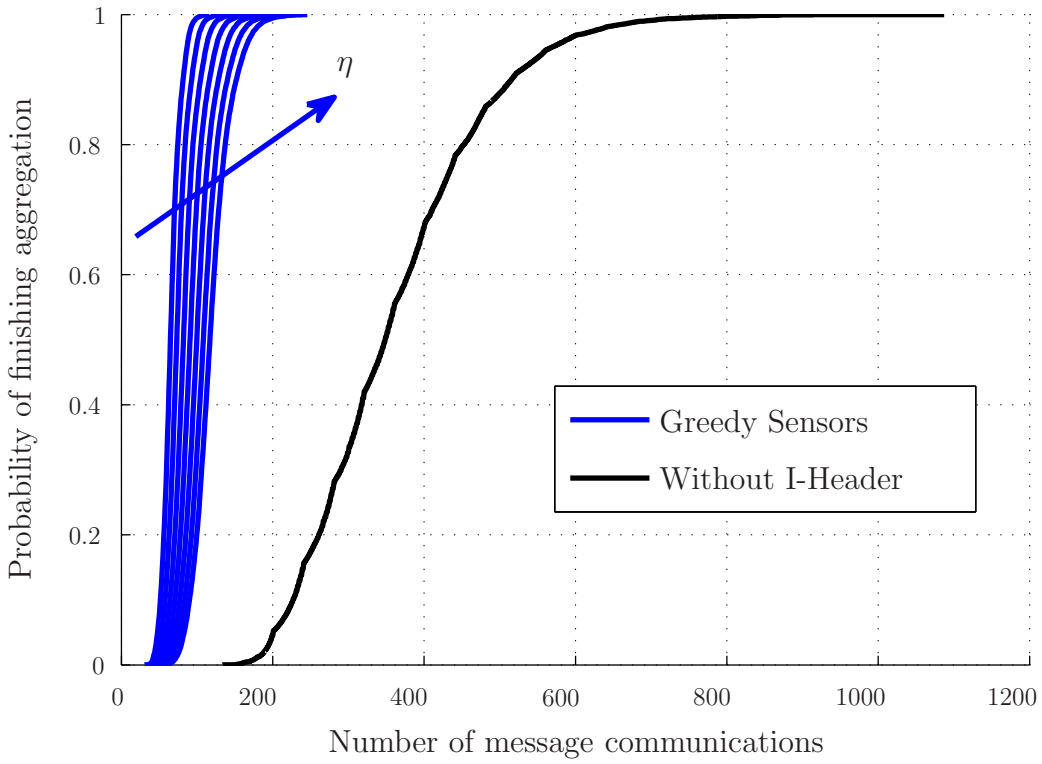

Figure 4.16. Impact of indicating headers with greedy sensors. From left to right, $\eta = 0\%, 5\%, 10\%, 15\%, 20\%, 25\%, 30\%$

number of required message communications. A significant reduction of the number of message communications can be witnessed by increasing the coordination depth from $c_i = 1$ corresponding to the scheme in [CKK13b] to $c_i = 2$. For coordination depths $c_i > 3$, the additional reduction by further increasing the maximum coordination depth is small. When sensors in the network are greedy sensors, i.e., a tree is constructed when a sensor wakes up, the performance of the number of message communications is shown in Fig. 4.20. In comparison to humble sensors, fewer message communications are needed with the greedy sensor strategy.

To consider the communications that have to be performed to exchange indicating headers, we assume that the indicating header requires 10% of the message length. We define *equivalent communications* as the sum of the number of message communications and 0.1 times the number of communications for indicating headers. The performance of the humble sensor case is shown in Fig. 4.21 and of the greedy sensor case is shown in Figure 4.22, respectively. As shown in Figure 4.21, with the humble sensor strategy, a larger coordination depth still results in a lower number of communications. However, with the greedy sensor strategy, such benefit by increasing coordination depth can only

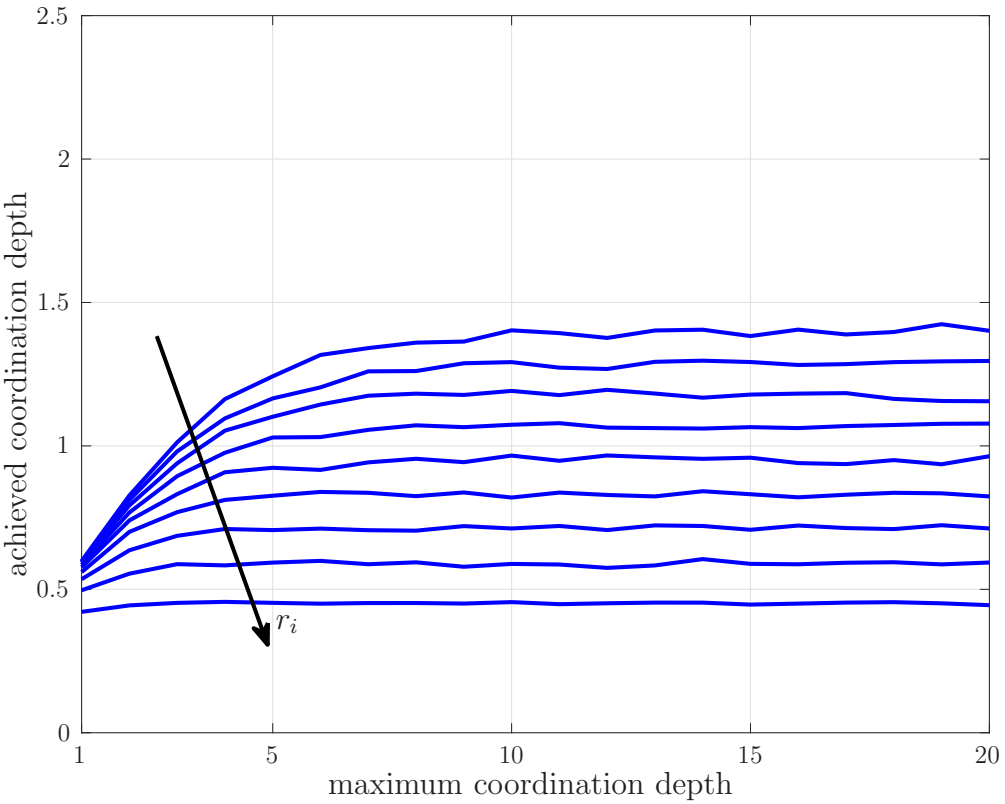

Figure 4.17. Maximum coordination depth versus achieved coordination depth with failure rate for a network with only humble sensors. Along the direction of the arrow, $r_i = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$

be achieved with a small failure rate $r_i < 0.3$. As seen in Figure 4.22, with a larger failure rate, the number of equivalent communications for a larger coordination depth is even worse compared to the case with smaller coordination depths.

4.5 Summary

This chapter discusses the reduction of the message communications in random gossiping with the support of the I-Headers introduced in Chapter 2. Except the connectivity, no constraints are given to the network topology. Therefore, the random gossiping applies also to a network with mobile sensors. Depending on how sensor exchanges messages with its neighbors, sensors are categorized into humble sensors and greedy sensors. The random gossiping algorithms utilizing I-Headers are discussed for both type of sensors. Furthermore, multihop coordination is introduced to increase

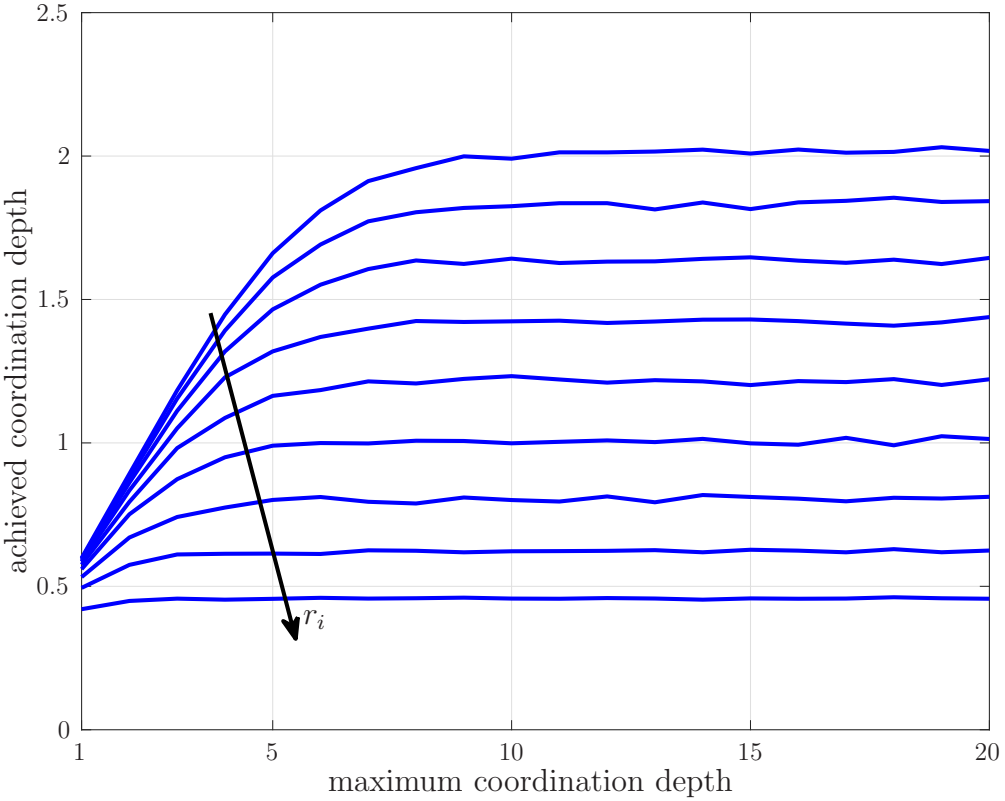

Figure 4.18. Maximum coordination depth versus achieved coordination depth with failure rate for a network with only greedy sensors. Along the direction of the arrow, $r_i = 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$

the speed of convergence by waking up more sensors at a time. In the simulations results, the significant reduction of the messages communications is observed by using the I-Headers in random gossiping. The further reduction in message communications can be seen by using multihop coordination.

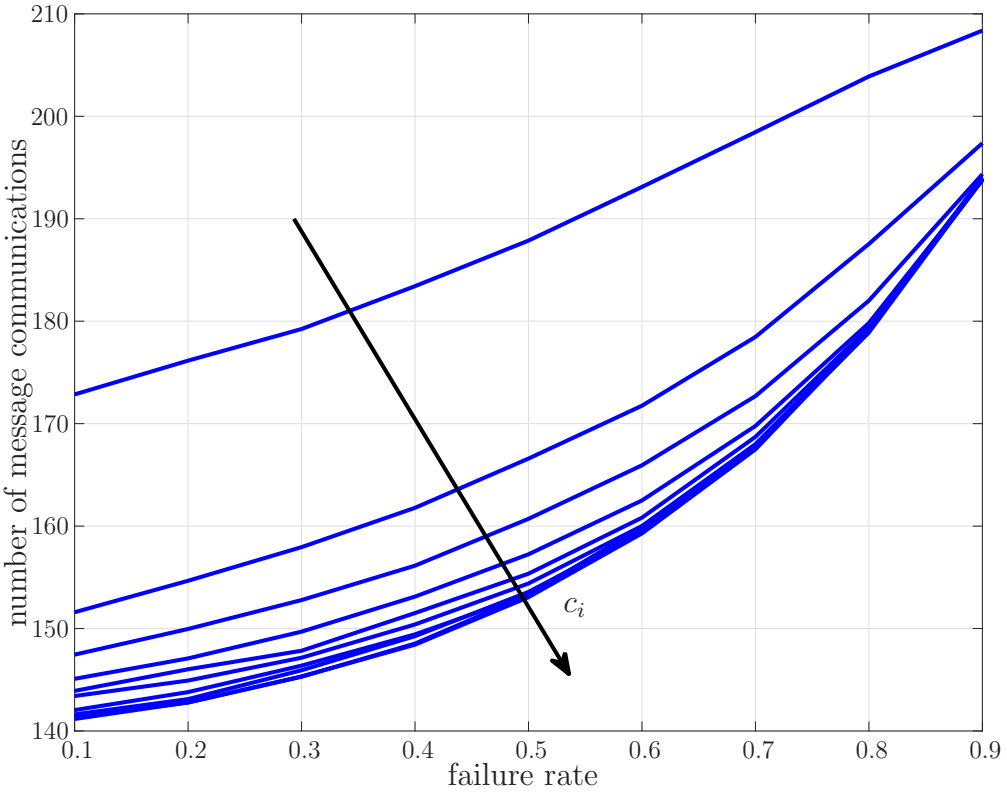

Figure 4.19. The average number of message communications required in the network until the aggregation is finished for humble sensors. Along the arrow, the maximum coordination depth $c_i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$.

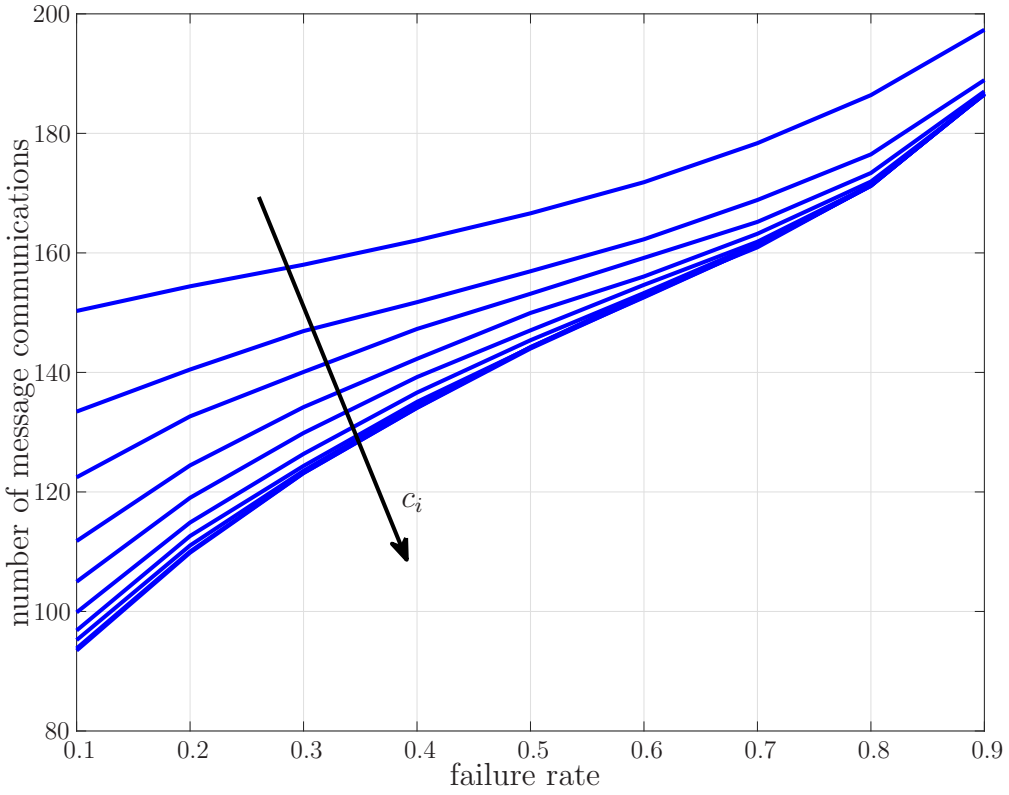

Figure 4.20. The average number of message communications required in the network until the aggregation is finished for greedy sensors. Along the arrow, the maximum coordination depth $c_i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$.

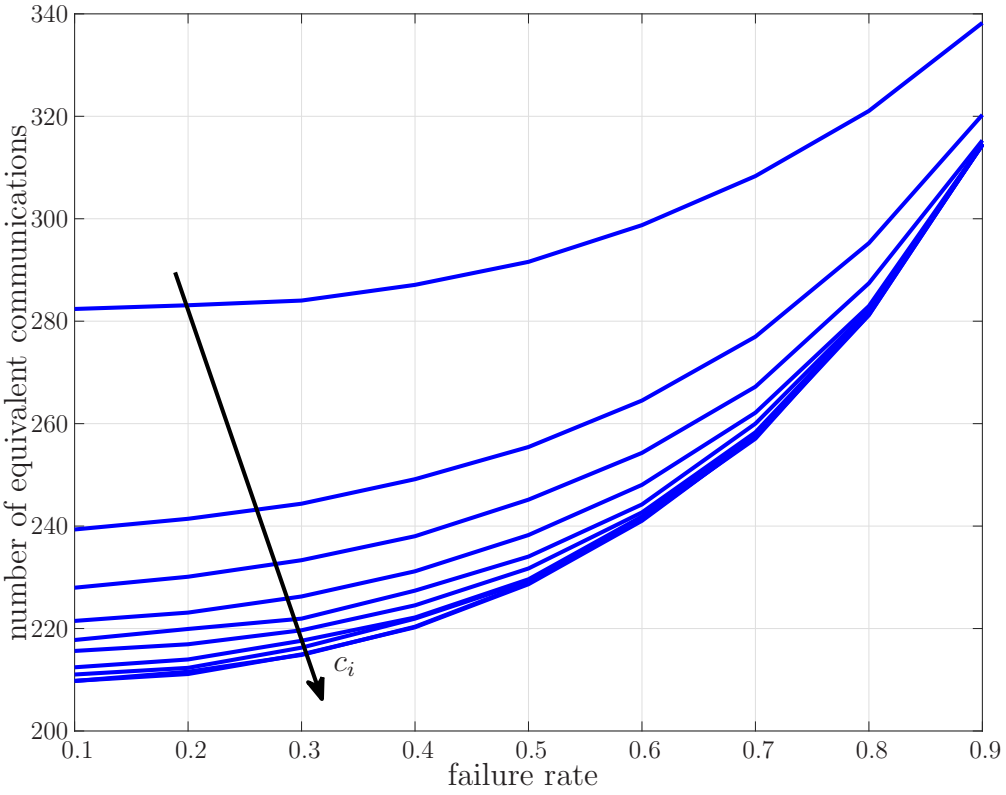

Figure 4.21. The average number of equivalent communications required in the network until the aggregation is finished for humble sensors. Along the arrow, the maximum coordination depths are $c_i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,$ respectively

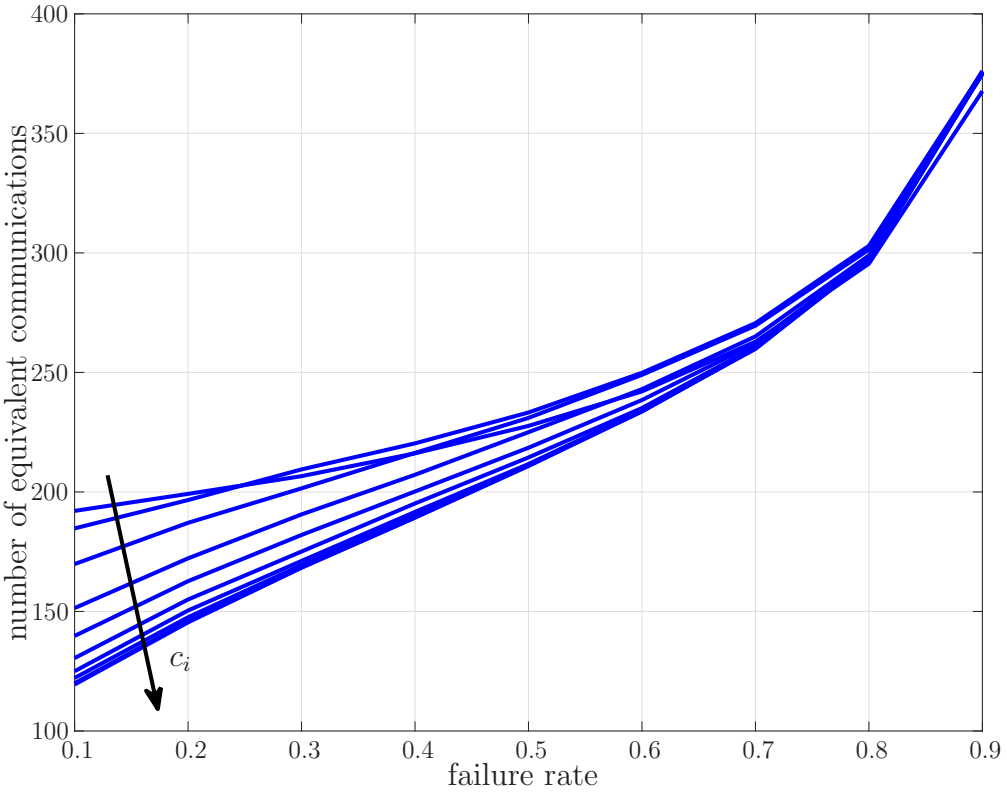

Figure 4.22. The average number of equivalent communications required in the network until the aggregation is finished for greedy sensors. Along the arrow, the maximum coordination depths are $c_i = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,$ respectively.

Chapter 5

Aggregation time reduction in static wireless sensor networks

5.1 Introduction

This chapter focuses on the further performance improvement of the random gossiping algorithms that we have introduced in Chapter 4 in static wireless sensor networks.

The random gossiping proposed in Chapter 4 uses I-Headers to determine whether message communications are necessary. A notable reduction in the number of message communications is witnessed. Under the assumption that the length of I-Header is much smaller than the length of the message, the equivalent message communication that takes the transmission of I-Header into account is also reduced significantly in comparison to the algorithm without using I-Header.

The discussion in this chapter considers a static constraint to the network topology of the wireless sensor networks. A wireless sensor network is static when the neighbor sensors of every sensor remain constant during the lifetime of the network.

Under the assumption of a static wireless sensor network, the proposed random gossiping algorithm can be further improved. Firstly, the number of I-Header communications can be reduced in static wireless sensor networks. This improvement is based on the knowledge of the I-Header of a sensor at its neighbor sensors, which is only available in static wireless sensor networks. A second improvement is achieved by introducing the idea of transmission deferment. Transmission deferment postpones the communications of sensors at specific topology locations such that more measurement data can be aggregated when they communicate with their neighbor sensors afterwards. Hence, the convergence speed is increased for the whole network. In this chapter, both topics will be discussed.

The remainder of the chapter is organized as follows. In Section 5.2, the I-Header reduction in static wireless sensor networks is discussed. Section 5.3 introduces transmission deferment. The performance and the conclusion are given in Section 5.4 and Section 5.5. Parts of the content of this chapter have been published by the author of this thesis in [CKK14a].

5.2 I-Header transmission reduction in static wireless sensor networks

By introducing I-Headers to the random gossiping, sensors compare the I-Header it receives from other sensors to detect the amount of new data that is contained in the message of other sensors. The comparisons result in the reduction of the message communications which contributes no new data to receiver sensors.

The cost of the reduction of message communications is a large number of communications to exchange I-Headers between sensors, especially in the random gossiping with greedy sensors. An example is shown in Figure 5.1, the I-Header at each sensor is given in the brackets. Two message communications are performed among the sensors with three I-Header communications.

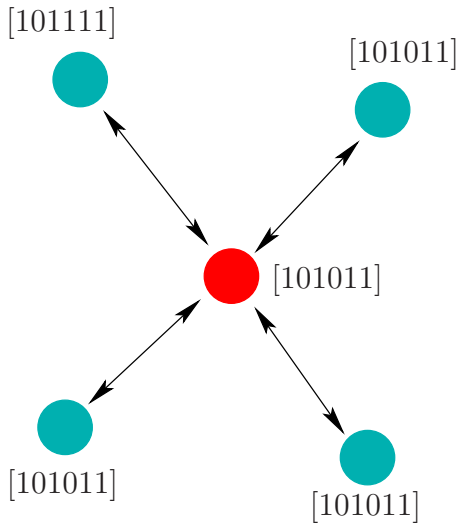

Figure 5.1. Demonstration of large I-Header communications

An extreme case would be that all sensors in the figures have the same I-Header. Even though no message communications are needed in this case, one I-Header shall be transmitted whenever the center sensor wakes up and initiates the communications.

The unawareness of the data in the messages of sensors in the neighbor is the main reason that I-Header exchanges are necessary. The random gossiping algorithm in Chapter 4 allows changes of neighbor sensors of a sensor. Therefore, the I-Headers are always communicated in order to choose the neighbor sensor to exchange their messages and to perform bias-cancellation. Figure 5.2 shows an example of a wireless sensor network where a "bottleneck" exists between two sub-networks. The sensor

v_i is the "bottleneck" which bridges two sub-networks. Sensors in each of the two sub-networks have aggregated messages with the same I-Header. Therefore, before the wake-ups of the bottleneck sensor initiating the communications, only I-Headers are exchanged among sensors, and no messages are communicated.

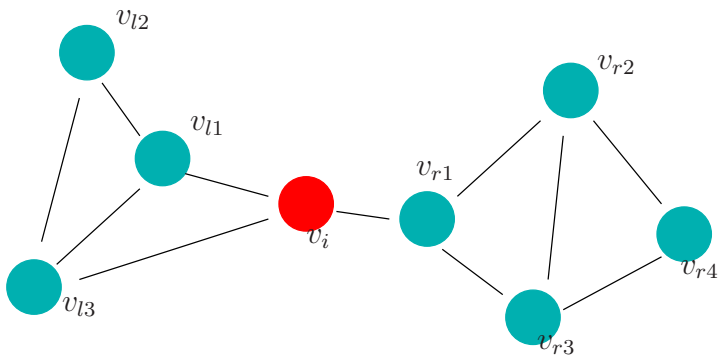

Figure 5.2. Demonstration of large I-Header communications with two sub-networks

In the two sub-networks, I-Headers are exchanged because sensors need to continuously check if there are new data in the messages of the neighbor sensors even though no new data has been aggregated.

Depending on how a sensor v_i aggregates a new measurement data, two types of communications are categorized in order to control the behavior of the sensor v_i differently in the considered algorithm.

- In a type-one communication, sensor v_i wakes up and initiates the communications with its neighbor sensors, i.e., sensor v_i communicates with its neighbor sensors as a center. Sensor v_i may aggregate new data by receiving messages from its neighbor sensors. A message broadcast is done later by v_i. Therefore, any new data aggregated by the v_i will eventually be aggregated by all its neighbor sensors.

  Sensor v_i performs twice I-Header communications during a wake-up in the type-one communication. In the first time, it broadcasts the I-Header to all its neighbor sensors to inform all its neighbor sensors the measurement data it has aggregated in the message. In the second time, v_i broadcasts its I-Header after it aggregates data from the messages transmitted by its neighbor sensors.

- In a type-two communication, sensor v_i is waked up as a neighbor sensor. In this type, v_i may aggregate new data by receiving the broadcast message from

the center. Therefore, only the center sensor, which is one neighbor sensor of v_i, knows the I-Header of v_i.

Sensor v_i performs at most only once the I-Header transmission during a wake-up in a type-two communication. After v_i received the I-header from the center sensor, it sends its I-Header to the center if it has aggregated new data to the center sensor.

Since the neighbor sensors of a sensor do not change in a static wireless sensor network, the sensor can use its buffer to store the I-Headers of its neighbor sensors which have been received from a previous I-Header communication. The stored I-Headers can be reused at the sensor when it communicates with its neighbor sensors, and its neighbor sensors have not aggregated any new data since their last communications.

With the categorization of the two types of communications and the capability of a sensor storing I-Headers of its neighbor sensors, I-Header communications can be reduced in static wireless sensor networks for both types of communications in three aspects.

- In the random gossiping, the center sensor (it is in a type-one communication) wakes up and initiates communications with its neighbor sensor. The first broadcasting of the I-Header of the center can be avoided in its type-one communications if a sensor has not aggregated any new data since its last type-one communications. In terms of avoiding an I-Header communication, neighbor sensors use the I-Header received in the last I-Header broadcasting of the center from their buffer.
- Neighbor sensors which are involved in type-two communications receive or recover the I-Header from the center. By comparison with the I-Header of their own messages, I-Header transmission from the neighbor sensors to the center can be avoided if their messages contain no new data to the center.
- After the center sensor aggregating the new data from the messages of the neighbor sensors, the center sensor can decide whether an I-Header broadcast is necessary by comparing its new I-Header and the I-Headers of its neighbor sensors. If the new I-Header at the center sensor contains no new data to the neighbor sensors, the transmission is omitted by the center sensor.

In all three cases, communications of I-Headers can be avoided. However, a message indicating that I-Headers are to be transmitted or to be recovered still need to be exchanged. This message exchange can be realized mainly in two ways:

- to transmit a 1-bit message telling the receiver that the I-Header shall be covered or will be transmitted, or
- to build an agreement between the transmitter and the receiver that at the particular phase of the random gossiping, if the receiver does not receive any message in a given period of time a recover of I-Header is performed at the receiver.

The random gossiping algorithm with the reduction of I-Header communications in static wireless sensor networks is given in Algorithm 10.

Algorithm 10 Random gossiping algorithm with I-Header reduction

v_i initiates communications with its neighbor sensors in $\mathcal{N}_i$;
v_i informs sensors in $\mathcal{N}_i$ whether it has new data updated in the data set of $\mathcal{S}_i$ of v_i since last type-one communication;
if there is new data updated in the data set of $\mathcal{S}_i$ of v_i since last type one communication **then**
    v_i broadcasts its I-Header $\boldsymbol{I}_i$ to all its neighbor sensors in $\mathcal{N}_i$;
else
    Sensors in $\mathcal{N}_i$ recover I-Header $\boldsymbol{I}_i$ that is received from the previous type-one communication initiated by v_i;
end if
for sensor v_j in neighbor sensors $\mathcal{N}_i$ **do**
    if message of v_j contains data that is new to v_i **then**
        v_j feed backs its $\boldsymbol{I}_j$ to v_i;
    else
        if message of v_i contains data that is new to v_j **then**
            v_j informs v_i that it requires message communication from v_i;
        else
            v_j transmits nothing;
        end if
    end if
end for
v_i processes all feedbacks;
v_i informs sensors in $\mathcal{N}_i$ which sensors need to send their messages;
All informed neighbor sensors transmit their messages;
v_i computes the new aggregated data;
if new aggregated data at v_i contains new data for sensors in $\mathcal{N}_i$ **then**
    v_i broadcasts the new I-Header $\boldsymbol{I}_i$;
    v_i broadcasts the new message;
end if

When all the sensors in the static wireless sensor networks have the same I-Header, the network is converged.

5.3 Transmission deferment

Transmission deferment is a state of the sensor in which the sensor defers its message communications. The goal of introducing transmission deferment is to reduce the total number of message communications in a static wireless sensor network.

We show an example using a network with 9 sensors in total depicted in Figure 5.3. There exists a bottleneck in the network at sensor v_i, and therefore, the network can be seen as a joint of two sub-networks. Using the random gossiping protocol we have introduced in Chapter 4, the simulation shows that in average 15 message communications are needed for the whole network to achieve convergence.

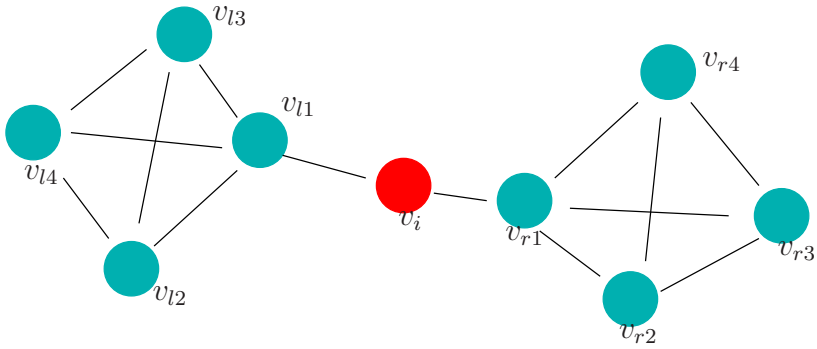

Figure 5.3. Example of a network with 9 sensors

In comparison, if sensor v_i defers its message communication until sensors in both sub-networks connected by sensor v_i have achieved local convergence of the messages, i.e., the same I-Header at the four sensors in each sub-network, in total 13 message communications are needed in the network instead of 15 message communications on average without transmission deferment. In this example, transmission deferment at sensor v_i reduces the number of message transmissions.

In order to develop a generic protocol of random gossiping with transmission deferment, it is assumed that when a sensor is in the state of transmission deferment,

- it can be woken up by other sensors. However, it shall only receive I-Headers, and not transmit I-Headers nor communicate messages, and
- it will not wake up as a center sensor.

When a sensor is in transmission deferment, a deferment criterion needs to be checked to determine whether the sensor shall remain in the state of transmission deferment.

When a criterion is met at the sensor that is in transmission deferment state, the sensor shall be able to wake up as a center sensor and to initiate messages communications with its neighbor sensors. The following two possible criteria are given as examples:

- I-Header matching: The deferment stops when the sensor receives from its neighbor sensors one or more I-Headers that match a pre-defined criterion, e.g., the received I-Header is the same to one I-Header of a defined I-Header list, or the received I-Header indicates that a pre-defined number of measurement data has been aggregated in the message of that neighbor sensor.
- Wake counting: The deferment stops when the sensor has been woken up as a neighbor sensor of other sensors for a given number of times.

In the example of Figure 5.3, the number of message communications reduced by introducing the transmission deferment at sensor v_i using the I-Header matching where the pre-defined I-Header list contains two I-Headers equalling to the I-Headers of the two converged sub-networks.

In addition to the deferment criteria, the condition is needed to determine which sensors shall enter the state of transmission deferment.

We use the sensor network in Figure 5.3 as an example to address the condition. In the network, sensor v_i is a bottleneck connecting two sub-networks, and the number of neighbor sensors of sensor v_i is 2. Let $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$ denote the I-Headers that sensor v_i will receive from its two neighbor sensors when the two sub-networks achieve their local convergence. The correlation of $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$, denoted by $\mathrm{Corr}(\boldsymbol{I}_1, \boldsymbol{I}_2) = \boldsymbol{I}_1(1) * \boldsymbol{I}_2(1) + \boldsymbol{I}_1(2) * \boldsymbol{I}_2(2) + ... + \boldsymbol{I}_1(N) * \boldsymbol{I}_2(N)$, will be zero. In this example, the number N is given by $N = 9$.

In order to consider the time used for the two sub-networks to achieve the local convergence in the correlation function, the subscript n is introduced to the correlation function. $\mathrm{Corr}_n(\boldsymbol{I}_1, \boldsymbol{I}_2)$ will indicate the correlation function of the two I-Headers that the red sensor receives after its n wake-ups. The correlation of $\boldsymbol{I}_1$ and $\boldsymbol{I}_2$ after infinite wake-ups of the red sensor is given as $\lim_{n\to\infty} \mathrm{Corr}_n(\boldsymbol{I}_1, \boldsymbol{I}_2) = 0$.

If sensor v_i sends its own I-Header to both of its neighbor sensors as a trigger of I-Header communications of the two sub-networks, the correlation result will be $\lim_{n\to\infty} \mathrm{Corr}_n(\boldsymbol{I}_1, \boldsymbol{I}_2) = 1$.

This condition can be checked at the same time for several sensors in a static network as long as these sensors have not exchanged the I-Headers received from those connected sub-networks.

In a more practical situation, a sensor can enter the state of transmission deferment when its number of neighbor sensors is N^{df} and after n^{df} wake ups, the maximum correlation function of the I-Headers received from any two neighbor sensors shall be C^{df}. These conditions can be abbreviated as $(N^{\mathrm{df}}, n^{\mathrm{df}}, C^{\mathrm{df}})$-condition for a sensor entering transmission deferment state. The condition for sensor v_i in the given example is $(2, 0, 0)$. In a static wireless sensor network, the condition can be determined based on analyzing the network and find the bottleneck sensor that may connect two or more sub-networks.

5.4 Performance and discussion

In the simulations, the number of sensors in the network is set to $N = 50$. These 50 sensors are randomly deployed in a squared area and the communication range is adjusted with a value to guarantee the connectivity of the network.

Figure 5.4 compares the performance improvement of I-Header transmission reduction for the sensors in the network in one WSN realization. The values of the black dots are the numbers of I-Header transmission performed by the sensors without Algorithm 10. The red dots give the number after using the algorithm. The results show the reduction of I-Header transmission observed at the individual sensor.

In Figure 5.5, the dashed lines show the average number of I-Header transmissions per sensor without the I-Header transmission reduction, whereas the solid lines depict the number of I-Header transmissions with the proposed algorithm. When the communication range is increased, the gain by using the proposed algorithm decreases since a larger communication range leads to faster convergence of the network, and the number of I-Header transmissions reduces for both schemes. The performance of both cases merges when the communication range reaches the value with which a sensor can communicate directly to all other sensors in the network.

In Figure 5.6, we show the reduced bias in the given WSN realizations with the introduced bias reduction algorithm purposed in Chapter 3. At maximum 14 bias (total number of duplicated measurement data aggregated) per sensor on average can be

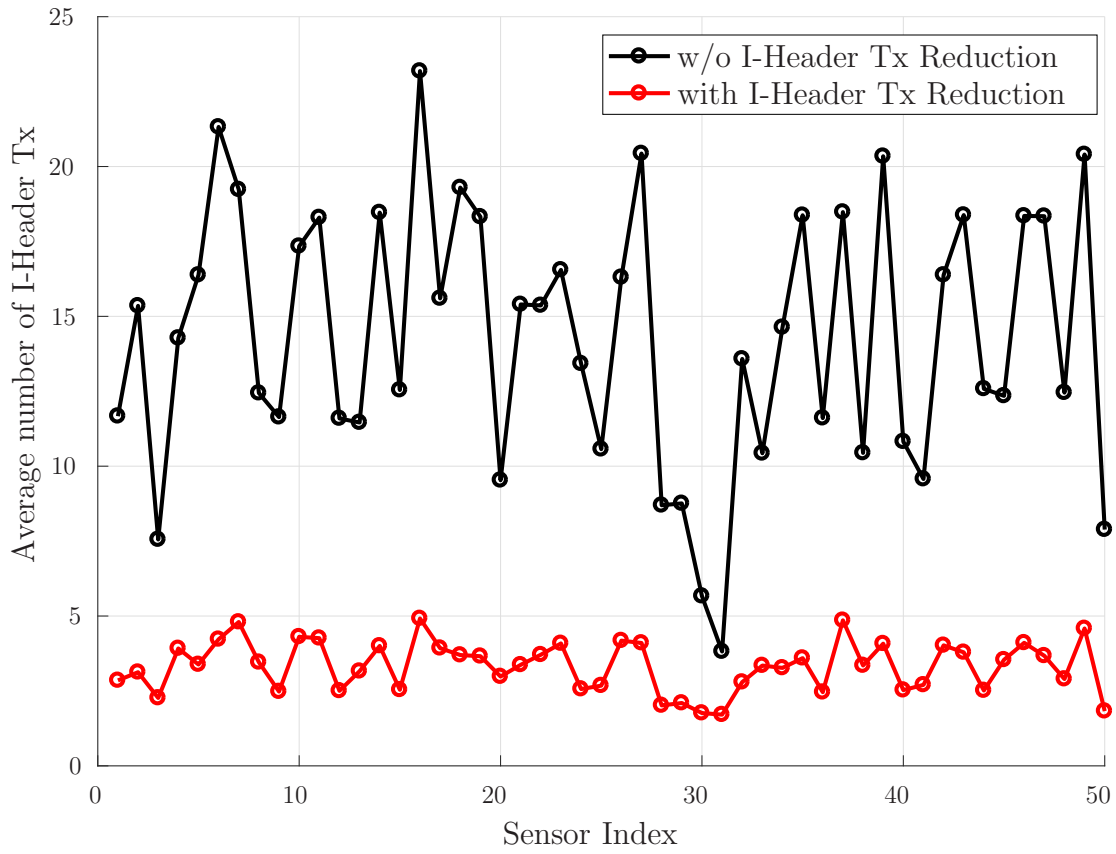

Figure 5.4. Reduction of I-Headers Communications

reduced. When the communication range is increased such that each sensor can communicate directly to all other sensors, there is no bias reduction observed since it requires only one gossiping communication for all sensor aggregate the measurement of the entire network.

5.5 Summary

In this chapter, we consider wireless sensor networks where a static constraint on the network topology is made. In the network, the neighbor sensors of every sensor remain constant in the network lifetime. Firstly, algorithms are given to reduce the necessity of I-Header communications by enabling sensors to remember the I-Headers of their neighbor sensors. Secondly, the transmission deferment is discussed. It explores a result that postponing a message communication of specific sensors at topologically "bottleneck" locations can reduce the total message communications in the network to achieve convergence.

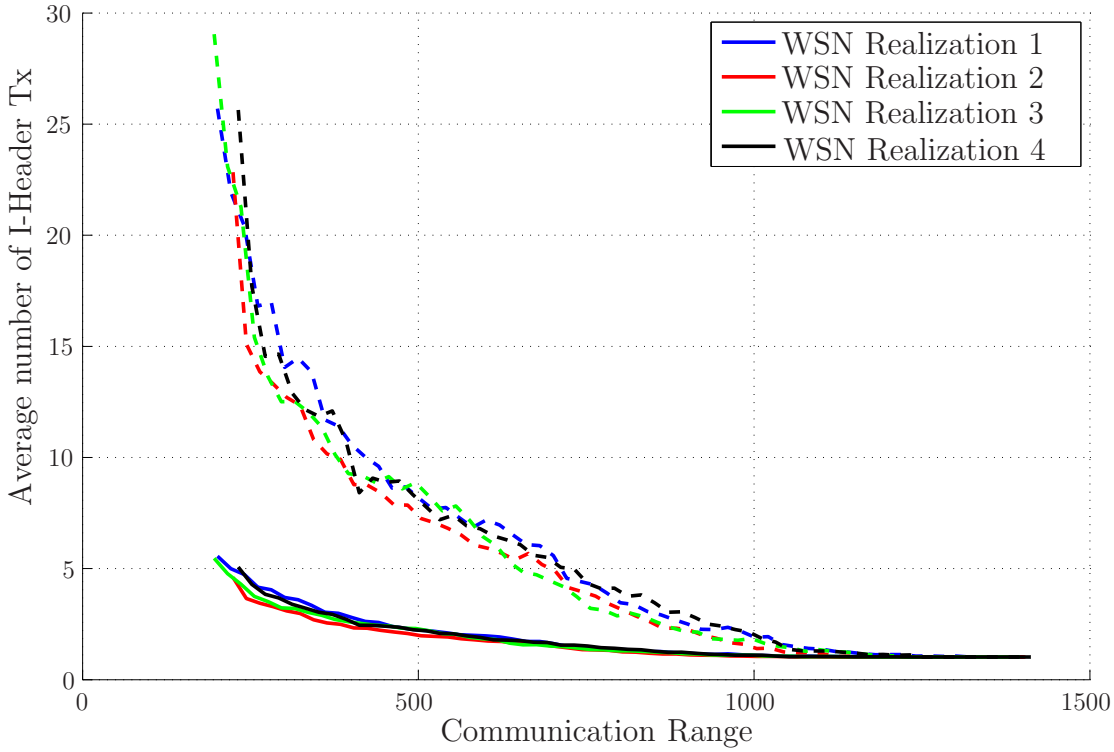

Figure 5.5. Number of I-Header communications vs. communication range

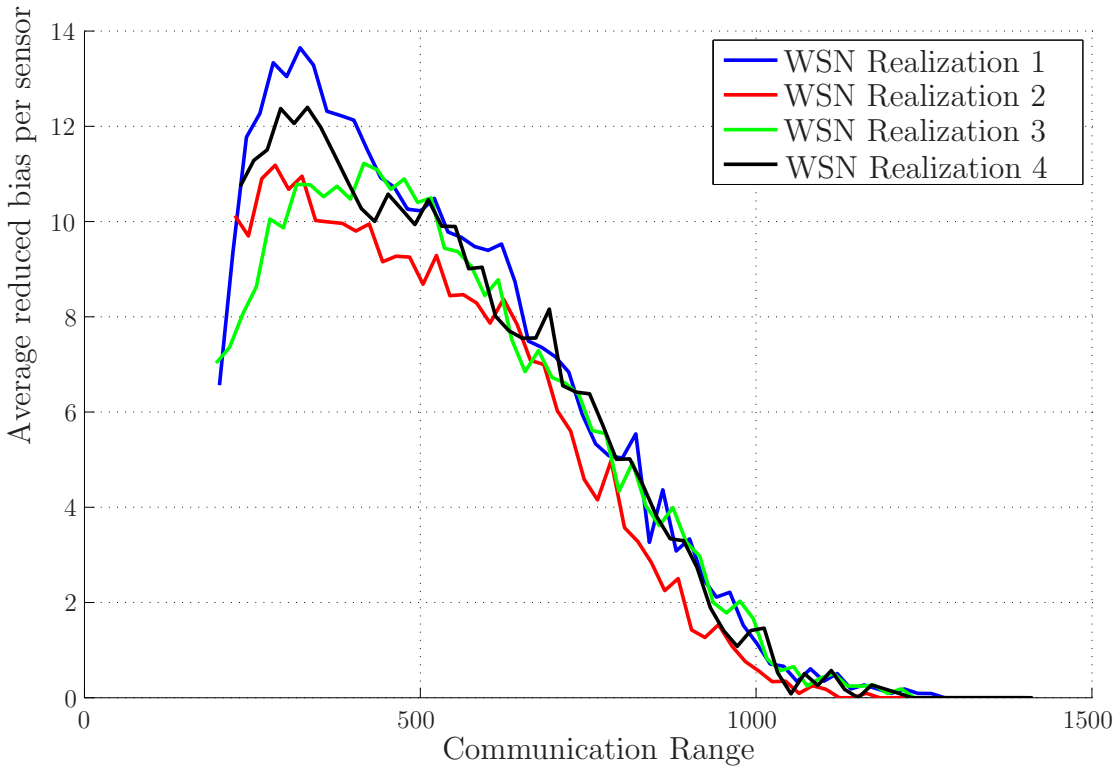

Figure 5.6. Reduced bias by applying the bias reduction algorithm

Chapter 6

Random gossiping refinement with partial application involvement

6.1 Introduction

In previous chapters, a refined random gossiping protocol is applied in wireless sensor networks without central scheduling of the communications. The objective of the refined random gossiping protocol is to aggregate the measurement data generated by all sensors and to have the aggregation result available at all sensors. If an application is defined by its type of measurement data and the computation that applies to the measurement data, sensors we have considered in the previous chapters run only one application.

When multiple applications are deployed in the network, one application may only involve a part of sensors. In this chapter, partial application involvement means that only a part of sensors in the network generate measurement data, perform computing with these data, and are interested in the final computing result[CKK14b]. The convergence is defined for an application in this case when all the sensors that are involved in an application have aggregated all the measurement data that are generated by them.

The partial application involvement may apply to the following two situations.

1. The sensors are densely deployed in an area where each position is covered on average by multiple sensors. Therefore, it does not require all sensor to perform sensing and computation for one application.
2. The sensor networks are sliced into subsets where each subset supports one application, exclusively.

We name those sensors that are involved in one application as "application member sensors", and the sensors that are not involved in that application as "non-application member sensors. In both of the situations as mentioned above, if the application member sensors are distributed in the network without being able to be directly connected, it results in a disconnected network for the application member sensors. The connectivity of the application member sensors can only be maintained by non-application

member sensors assisting the communications. When assisting the communications between application member sensors, the non-application member sensor will have to transmit and receive messages. However, no computations will be performed, which is different than the random gossiping that has been discussed in the previous chapter.

Due to this difference, a further refinement of the random gossiping is needed to enable both application member sensor and non-application member sensor being co-existed in one network. In this chapter, we propose the refinement by using the concept of I-Header introduced in Chapter 2 to the random gossiping that has been discussed in chapter 4. The targets of the refinement are to minimize

- the number of non-application member sensors that have to assist the communications between the application member sensors, and
- the number of communications performed by non-application member sensors.

Meanwhile, the total number of communications performed in the network shall not be increased with the proposed refinement.

When using the I-Headers, sensors will treat the bits that correspond to the non-application member sensors as "irrelevant". During the neighbor discovery phase, a sensor can know whether another sensor in its neighbor is an application member sensor or it is a non-application member sensor.

In Section 6.2 we discuss the categorizations of the two sensors and the corresponding scenarios of the refined random gossiping. The detail of the refined random gossiping is introduced in Section 6.3. Part of the content of this chapter has been published by the author of this thesis in [CKK14b].

6.2 Scenario categorization

For a given application in a wireless sensor network, the total sensors can be divided into two subsets. The set $\mathcal{V}_{\mathrm{A}} \subset \mathcal{V}$ includes the sensors that are involved in the application (application-member sensors), and the set $\mathcal{V}_{\mathrm{K}} = \mathcal{V} \backslash \mathcal{V}_{\mathrm{A}}$ denote the sensors which are not involved in the applications (non-application-member sensors). Let $\mathcal{V}_{\mathrm{B}} \subseteq \mathcal{V}_{\mathrm{K}}$ denote the non-application-member sensors which assist in the message communications. Let N_{A}, N_{K}, and N_{B} denote the number of sensors in $\mathcal{V}_{\mathrm{A}}$, $\mathcal{V}_{\mathrm{K}}$, and $\mathcal{V}_{\mathrm{B}}$, respectively. The

ratio of N_{A}/N is denoted by η_{A}, i.e., the number of application-member sensors to the total number of sensors.

Furthermore, let T_{A} and T_{B} denote the number of communications performed by sensors in $\mathcal{V}_{\mathrm{A}}$ and $\mathcal{V}_{\mathrm{B}}$ until the two objectives mentioned in Chapter 2 are achieved, respectively, where $T = T_{\mathrm{A}} + T_{\mathrm{B}}$ is the total number of communications.

In the refined random gossiping, it shall be ensured that T_{A} is small such that the application-member sensors can quickly have the computation output. Moreover, N_{B} and T_{B} shall also be small such that only a few non-application-member sensors are assisting in the communications performing only a limited number of communications.

Unlike the random gossiping discussed in previous chapters where sensors communicate with their neighbor sensors and that whether a message shall be exchanged is purely based on algorithms involving I-Headers, the refined random gossiping in this chapter shall consider one more dimension of the problem: whether a sensor and its neighbor sensors are application-member or non-application-member sensors. According to the type of sensor which initiates the communications (center) and the types of its neighbor sensors, we categorize our problem into six scenarios, as shown in Table 6.1.

Table 6.1. Scenarios categorization

	Center sensor	Neighbor sensors
Scenario 1	application-member	application-member
Scenario 2	application-member	application-member and non-application-member
Scenario 3	application-member	non-application-member
Scenario 4	non-application-member	application-member
Scenario 5	non-application-member	application-member and non-application-member
Scenario 6	non-application-member	non-application-member

- Scenario 1: In the first scenario, the center sensor, as well as all its neighbor sensors, are application member sensors, as shown in Figure 6.1.

  In this scenario, there are no non-application-member sensors involved. Therefore, the random gossiping protocol discussed in the previous chapter can be used.

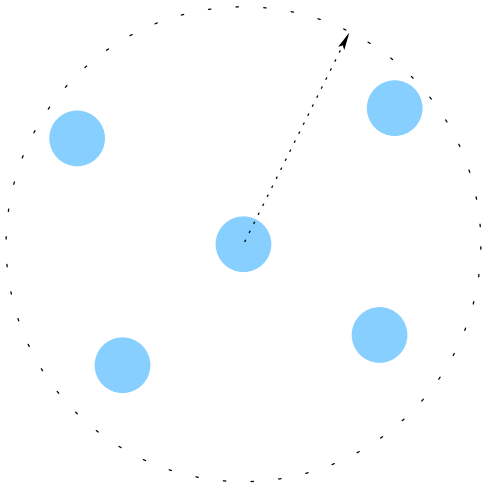

Figure 6.1. Scenario 1

- Scenario 2: The second scenario is shown in Figure 6.2 where the center sensor is an application member sensor (solid circles), and the neighbor sensors are partly application-member sensors (solid circles) and partly non-application-member sensors (white circle).

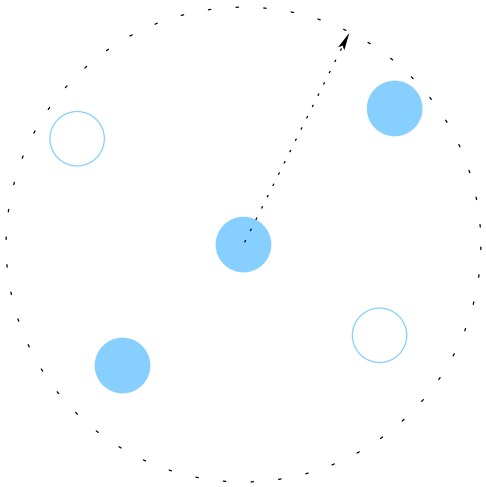

Figure 6.2. Scenario 2, solid circles are application-member sensors, and white circles are non-application-member sensors

To design the protocol in this scenario, we define four disjoint conditions that may happen for the center sensor.

a The center sensor and its application-member neighbor sensors have messages to exchange.

b The center sensor and its application-member neighbor sensors have the

same message. Therefore, no message exchange is needed between them.

c The non-application-member neighbor sensors have application messages to transmit.

d The non-application-member neighbor sensors have no application messages to transmit.

The conditions a and b determine whether the message communication between the application-member center sensor and the application-member neighbor sensors should be performed. Respectively, the following two questions need to be answered.

1. How should the center application-member sensor communicate with its application-member neighbor sensors?
2. Should the communication between the center application-member sensor and its non-application-member neighbor sensors take place at the same time?

The center sensor and its application-member neighbor sensors all generate data and all interest in the data aggregation. Therefore, the random gossiping protocol discussed in previous chapters can be applied. For the non-application-member neighbor sensors, the objective is to minimize the number of communications performed by them. To achieve this in this scenario, the best strategy to offer is to wait until all reachable application-member sensors around it have finished the data aggregation locally with their own application-member neighbor sensor. Then the non-application-member sensor only helps the message exchanges among several groups of application-member sensors instead of every individual application-member sensor in its neighbor. Therefore, the communication between the center application-member sensor and its non-application-member neighbor sensors should not be simultaneous.

The communication between the center sensor and the non-application-member neighbor sensors consists of two more problems:

3. Should the application-member center sensor initiate the communication between the center application-member sensor and the non-application-member neighbor sensors?
4. Will the protocol be different for the case that the non-application-member sensors have application messages received from other application-member sensors or non-application-member sensors and the case that the non-application-member sensors have no messages but can receive messages from the application-member center sensor?

If the application-member center sensor initiates the communication when the non-application-member neighbor sensors have application messages, the center sensor and its application-member neighbor sensors are able to aggregate more data with fewer communications. It is because that the message from the non-application-member neighbor sensors can transmit the message to the center sensor and then the center sensor can broadcast to all neighbor sensors. However, the non-application-member sensor may need to perform communications of the same message considering that there may be other application-member sensors as its neighbors. This will be revealed in the following scenarios.

If the non-application-member neighbor sensors have no messages, and the center initiates the communications with them, messages will be sent to these non-application-member neighbor sensors because of the broadcast. This can lead to the problems that the non-application-member neighbor sensors may have to transmit a significant number of messages received from the application-member sensors, and all those messages should first be aggregated by application-member sensors to reduce the times that non-application-member sensors receive and transmit.

A solution would be a complete role changing concerning the communication initiation. The message transmission between non-application-member sensors and the application-member sensors is initiated by non-application-member sensors when they are the center sensors, as discussed in the following scenarios. Furthermore, problems such as when and how the non-application-member sensors initiate the communications will also be addressed.

- Scenario 3: In the third scenario, the center sensor is an application-member sensor, and all its neighbor sensors are non-application-member neighbor sensors, as shown in Figure 6.3.

  In this scenario, the role played by non-application-member neighbor sensors is clearly and intuitively to see. As a solely application-member neighbor sensor, it requires the assistance of the non-application-member neighbor sensors to relay its message to other application-member sensors and receive messages from them. If a center application-member sensor initiates the communication, it may greedily rely on its non-application-member neighbor sensors to communicate messages (greedy sensor). This results in a large number of communications by the non-application-member sensors. Additionally, one or more of the non-application-member neighbor sensors may have no other neighbor sensors, and then they cannot help to further forward the message.

- Scenario 4: In this scenario as shown in 6.4, the center sensor is a non-application-

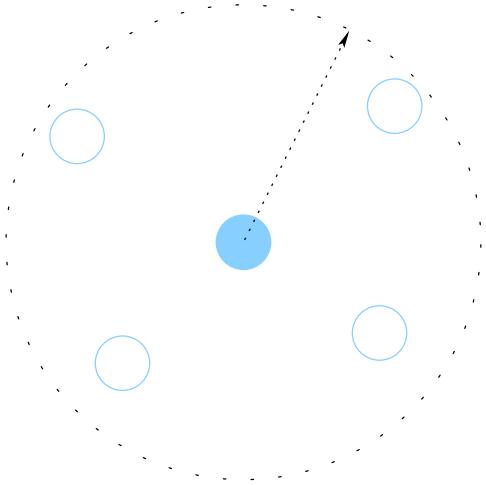

Figure 6.3. Scenario 3, solid circle is an application-member sensor, and white circles are non-application-member sensors

member sensor and all its neighbor sensors are application member neighbor sensors.

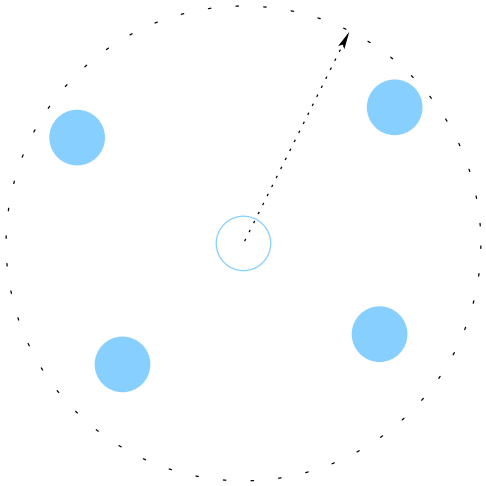

Figure 6.4. Scenario 4

To discuss how the center non-application-member sensor should involve in the communication, we consider two extreme cases. In the first case, all application-member neighbor sensors can reach each other. In this case, the center sensor should not involve in the communications since the message can be exchanged without the participation of the center sensor. In the second case, there is no connection between any two application-member sensors. Therefore, the center non-application-member sensor is the only "bridge" for them to exchange messages. As shown in these two cases, how to utilize the non-application-member

center sensor efficiently depends on the connectivity of its application-member neighbor sensors. However, as we stated in previous chapters, such connectivity information is not available at the sensor.

In order to avoid as much as possible the unnecessary messages communications performed by the center non-application-member sensors, we develop a protocol in the next section based on a so-called "request" which is transmitted from application-member sensors to non-application-member sensors.

- Scenario 5: Figure 6.5 shows scenario 5, where the center sensor is a non-application-member sensor, and the neighbor sensors are mixed with both application-member and non-application-member sensors.

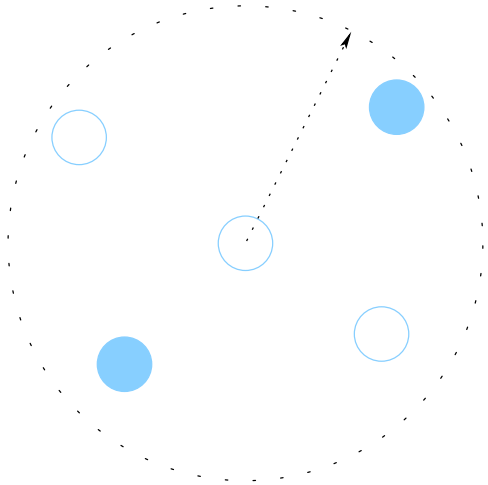

Figure 6.5. Scenario 5

As a center node, the non-application-member may play many roles:

  - The center non-application-member sensor can help its application-member neighbor sensors to exchange their messages of the application.
  - The center non-application-member sensor can forward the message from the application-member sensors to non-application-member sensors.
  - The center non-application-member sensor can forward the message from the non-application-member sensors to application-member sensors.

Each of the three roles turns out requiring an individual solution. For the first role, the center sensor can ignore the non-application-member sensors in its neighborhood and behave as what is done in scenario 5 to "bridge" messages exchanges among the application-member sensors. For the second role, the center sensor needs to negotiate with its non-application-member neighbor sensor to see if any

sensor is free to receive the message that is to be forwarded from the application-member sensor via the center sensor. In the third role, the center needs to help to attract the message from the non-application-member sensor to the application-member sensor if the message contains any new data to the application-member sensor. Moreover, arbitration shall be done when the roles can be played at the same time in one wake-up.

- Scenario 6: Last but not least, scenario 6 happens when the center sensor, as well as all its neighbor sensors, are non-application-member sensors, as shown in 6.6.

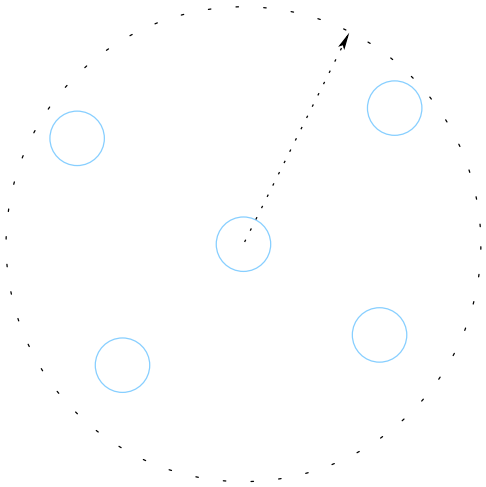

Figure 6.6. Scenario 6

The first problem in this scenario is to determine the condition that the non-application-member sensor should wake up as a center to initiate the communications. In general, there are only two states that happen at the non-application-member sensor.

  - It has an application message. In this state, the center intends to send its message out to its neighbor(s).
  - It does not have an application message. In this state, the only thing that a sensor can do is to forward the message from one of its non-application-member neighbor sensor to another non-application-member neighbor sensor. However, as we have discussed in scenario 2, a non-application-member sensor with an application message sends its message only when it wakes up as a center. The non-application-member sensor in this state will not wake-up as the center.

There is another problem in this scenario requires additional consideration when designing the protocol. If all sensors in this scenario, including the center and all

the neighbor sensors, have application messages, there will be a local deadlock and no message can be forwarded. It is because non-application-member sensors will not perform data aggregation. The solution can be to abandon the message or to buffer the message in the memory. By abandoning the messages, the center can behave as a sensor ready to help to forward the application messages. However, in the case that a message has been forwarded through a sequence of transmissions, to abandon the message implies that all the transmissions are wasted in terms of energy and channel resource. When to buffer the message and then to behave as a sensor ready for forwarding other application messages, it has to consider in the protocol of how to deal with the message being buffered, and when to transmit the buffered message.

In both Scenario 5 and Scenario 6, there is a common problem that the non-application-member sensor may transmit a message more than once or transmit a message which is "older" than its previously transmitted messages. Here "older" means that the message contains less or the same measurement data than the message which have been previously transmitted by this non-application-member sensor. In order to overcome this problem, we enable memory capability at non-application-member sensors. Different from the memory capability at the application-member sensor that both I-Header of the message and the message itself will be stored, a non-application-member sensor only stores I-Headers. The stored I-Headers are used to avoid transmitting the same message more than once and transmitting "older" messages by the non-application-member sensors. Details of how the non-application-member sensors store and reuse the I-Headers are revealed in the next chapter.

6.3 Refined random gossiping protocol

In this section, protocols for the six different scenarios discussed in the previous section are given. Meanwhile, the problems that arose in the six scenarios are solved.

As we introduced in Chapter 2, I-Header functions as a database shared across several layers for a cross-layer consideration. In Chapter 3 and Chapter 4, we see the potential that I-Header benefits the random gossiping in general function calculations in terms of reducing the bias and reducing the number of message communication. In this chapter, I-Header are used assist the communications not only between the application-member sensors but also between a non-application-member sensor and an application-member sensor as well as between non-application-member sensors.

6.3.1 Protocol for scenario 1

In Scenario 1, the center sensor $v_{\mathrm{AM},i}$, as well as the neighbor sensors $v_j \in \mathcal{N}_{\mathrm{AM},i}$, are all application-member sensors. As we discussed in the previous section, the random gossiping protocol discussed in Chapter 4 can be applied in this scenario. The interactions between the application-member sensors are based on I-Header communications and comparisons. Recall the definition of the r function from previous chapters, we have

$$r(\boldsymbol{I}_i, \boldsymbol{I}_j) = \begin{cases} 1 & \text{for } \mathcal{S}_i = \mathcal{S}_j \quad (6.1) \\ 2 & \text{for } \mathcal{S}_i \supset \mathcal{S}_j \quad (6.2) \\ 3 & \text{for } \mathcal{S}_i \subset \mathcal{S}_j \quad (6.3) \\ 4 & \text{for all else .} \quad (6.4) \end{cases}$$

In Algorithm 11, we assume the application-member sensor v_i wakes up as the center sensor.

Algorithm 11 has been modified to match the context of the partial sensor involvement and using the improved bias-cancellation discussed in Chapter 3.

6.3.2 Protocol for scenario 2

As have been discussed in the previous section, the application-member center sensor initiates the communications with all its application-member neighbor sensors firstly. The communications are guided using the protocol in Algorithm 11.

Since the message communications between non-application-member sensors and the application-member sensors shall be initiated by the former, a strategy shall be given about how and when the non-application-member sensors initiate the message communications. Therefore, we propose a concept called *request*. The request is always sent from application-member sensors to non-application-member sensors after the application-member sensor as a center finishes data aggregation with its neighbor application-member sensors, i.e., the I-Header are the same at the application-member sensor and all its application-member neighbor sensors, or in another word, the set of data aggregated in their applications messages are the same.

For every sensor $v_j \in \mathcal{N}^{\mathrm{K}}$, we define a counter z_j counting the number of received request from its application-member neighbor sensors. The protocol for Scenario 2 is given in Algorithm 12.

Algorithm 11 Protocol for Scenario 1

v_i initiates the communications with all sensors in $\mathcal{N}_i^{\mathrm{A}}$
v_i broadcast I-Header $\boldsymbol{I}_i$ to all sensors in $\mathcal{N}_i^{\mathrm{A}}$
$\forall v_j \in \mathcal{N}_i^{\mathrm{A}}$ compares received I-Header $\boldsymbol{I}_i$ with its own I-Header $\boldsymbol{I}_j$ and generates $r(\boldsymbol{I}_i, \boldsymbol{I}_j)$
if $r(\boldsymbol{I}_i, \boldsymbol{I}_j) \in \{2, 4\}$ **then**
    v_j transmits I-Header $\boldsymbol{I}_j$ to v_i
else
    if $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 3$ **then**
        v_j transmits information to v_i, indicating that message at v_i contains new data to v_j
    else
        v_j does nothing
    end if
end if
if v_i receives any I-Header **then**
    v_i performs neighbor sensor selection based on all received I-Headers according to (3.24), finding the subset of sensors $\mathcal{N}_i^{\mathrm{b}}$ and the bias-reduction set
    v_i informs sensors in $\mathcal{N}_i^{\mathrm{b}}$ transmitting their application message to v_i
    $\forall v_j \in \mathcal{N}_i^{\mathrm{b}}$ transmits their application messages to v_i
    v_i perform data aggregation and bias-cancellation generating a new message and a new I-Header $\boldsymbol{I}_i$
    v_i compares new I-Header $\boldsymbol{I}_i$ with the received I-Header, generating $r(\boldsymbol{I}_i, \boldsymbol{I}_j)$ for all v_j having sent their I-Headers to v_i
end if
if (v_i receives any I-Header **and** there exist $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 3$) **or** v_i received from any v_j indicating $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 3$ **then**
    v_i broadcasts its new I-Header $\boldsymbol{I}_i$
    v_i broadcasts its new application message
    $\forall v_j \in \mathcal{N}_i^{\mathrm{A}}$, v_j compares received I-Header $\boldsymbol{I}_i$ with its own I-Header $\boldsymbol{I}_j$ and generates $r(\boldsymbol{I}_i, \boldsymbol{I}_j)$
    if $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 3$ **then**
        v_j sets $\boldsymbol{I}_j = \boldsymbol{I}_i$ and replaces its application message with the received application message.
    end if
end if
v_i terminates the communications with $\mathcal{N}_i^{\mathrm{A}}$

Algorithm 12 Protocol for Scenario 2

Perform protocols given in Algorithm 11
v_i initiates communication with its non-application member neighbor sensors $\mathcal{N}_i^{\mathrm{K}}$
$\forall v_j \in \mathcal{N}_i^{\mathrm{K}}$, $z_j = z_j + 1$
v_i terminates communications with sensors in $\mathcal{N}_i^{\mathrm{K}}$

As shown above, the requests are aggregated at the non-application-member sensors when they are sent by application-member sensors. The action on how to utilize the aggregated requests by the non-application-member sensors are discussed in scenarios 4 and 5 when the non-application-member sensors wake up as center sensors.

6.3.3 Protocol for scenario 3

In Scenario 3, the center has only non-application-member neighbor sensors. When the center initiates the communication, it only broadcasts a *request* to its non-application-member neighbor sensors. The protocol for this scenario is given in Algorithm 13.

Algorithm 13 Protocol for Scenario 3

1: v_i initiates communication with its non-application member neighbor sensors $\mathcal{N}_i^{\mathrm{K}}$
2: $\forall v_j \in \mathcal{N}_i^{\mathrm{K}}$, $z_j = z_j + 1$
3: v_i terminates communications with sensors in $\mathcal{N}_i^{\mathrm{K}}$

6.3.4 Agency and feedback

Before we propose the protocols for Scenario 4, 5, and 6, we introduce two concepts to assist in building the protocols.

The first concept is the so-called *agency*. In a static wireless sensor network, a non-application-member sensor $v_{\mathrm{NAM},i} \in \mathcal{V}_{\mathrm{B}}$ can be an agency when

- $v_{\mathrm{NAM},i}$ has no application-message,
- $v_{\mathrm{NAM},i}$ has at least one application-member neighbor sensor,
- All application-member neighbor sensors have the same application message, and therefore have the same I-Header, and
- $v_{\mathrm{NAM},i}$ has at least one non-application-member neighbor sensor.

We use $a_i = 1$ to denote if sensor v_i is an agency sensor and $a_i = 0$ if is not. When sensor $v_{\mathrm{NAM},i}$ is an agency sensor, $v_{\mathrm{NAM},i}$ interacts with other non-application-member sensors by using I-Header $\boldsymbol{I}_i$.

In principle, the objective to introduce the concept of agency is to reduce the message communications that are performed by the non-application-member sensors. Agency benefits this objective by introducing an efficient communication strategy for every non-application-member sensor, which has both application-member neighbor sensors and non-application-member neighbor sensors.

The non-application-member sensors are involved in the communication to assist the message exchanges among application-member sensors. We can decompose this problem to that the non-application-member sensors should help application messages exchanges among its neighbor application-member sensors and that it helps all its neighbor application-member sensors to exchange application messages with other application-member sensors which can only be reached via other non-application-member sensors. Figure 6.7 shows a simple example.

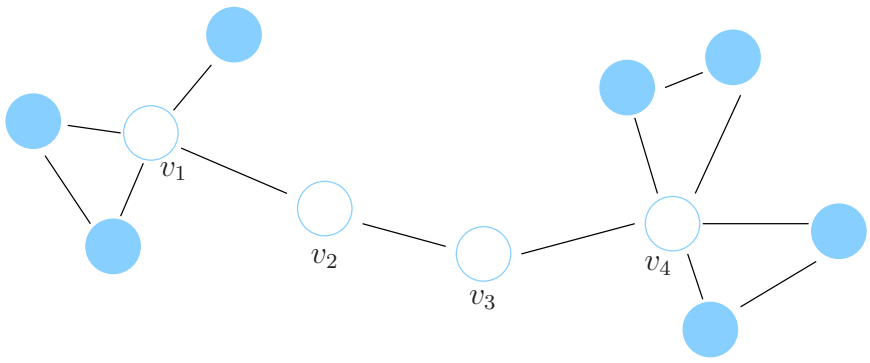

Figure 6.7. Example of agency

In Figure 6.7, non-application-member sensors v_1 and v_4 assist the application-member sensors in their neighbors for aggregation since the application-member sensors cannot be directly connected. In order to assist the message exchanges between application-member sensor in the neighbor of v_1 and v_4, other non-application-member sensors such as v_2 and v_3 are involved.

When the non-application-member sensor helps application messages exchanges among its neighbor application-member sensors, it forwards a message from one application-member neighbor sensor to other application-member neighbor sensors if necessary. In Scenario 5, we are dealing with this problem mainly. However, the non-application-member sensor in Scenario 5 cannot be an agency according to the definition. The function of an agency is to represent a set of application-member sensors which have the same application message and I-Header by non-application-member sensors to interact with other non-application-member sensors. From this point of view, the agency extends the range of interaction of application-member sensors by using their non-application-member neighbor sensors. In the example shown in Figure 6.7, sensor

v_1 is an agency for its neighboring application-member sensors to communicate the application messages with sensor v_2,

The limitation of using an agency is stated in its definition that it only works for static wireless sensor networks. This condition can be relaxed under the assumption that the network topology remains static during the time a non-application-member sensor becoming an agency and then becoming a non-agency sensor.

The second concept is *feedback*. Feedback is used to categorize the information that will be sent from neighbor sensors to the non-application-member center sensor for the center sensor to decide the protocol process.

In previous chapters, neighbor sensors may transmit their I-Header, their (application) message, and the information that they require the message from the center. The first two does not belong to the feedback that we are proposing here. When the center sensor v_i is a non-application-member sensor, there are two cases when it has an I-Header:

- v_i has an application message and therefore has an I-Header
- v_i is an agency, therefore it has an I-Header.

In both cases, when v_i wakes up and initiates the communications with its neighbor sensors, it can broadcast its I-Header to all its neighbor sensors. A feedback transmission is triggered by a neighbor sensor receiving an I-Header from the center non-application-member sensor. The categorization of the feedback depends

- whether the neighbor sensor is an application-member sensor or a non-application-member sensor,
- whether the neighbor sensor has an application message or has no application message,
- whether the neighbor sensor has already transmitted a message that includes all data referred in the received I-Header, and
- what the relation is between the received I-Header and its I-Header at the neighbor sensor.

To clarify the third bullet, let ε^{v_j} denote a function at the non-application-member neighbor sensor v_j of v_i taking the received I-Header $\boldsymbol{I}_i$ as a parameter. Function ε^{v_j} compares $\boldsymbol{I}_i$ to all the stored I-Headers $\boldsymbol{I}_l^{v_j} \in \Psi^{v_j}$ at sensor v_j and tells whether the data set $\mathcal{S}_i = \Theta^{-1}(\boldsymbol{I}_i)$ of the application message at sensor v_i contains new data to the data set $\mathcal{S}_l^{v_j} = \Theta^{-1}(\boldsymbol{I}_l^{v_j})$. If there is new data contained, i.e., $r(\boldsymbol{I}_j, \boldsymbol{I}_l^{v_i}) \in \{2, 4\}$, the result of the function is $\varepsilon^{v_j}(\boldsymbol{I}_i) = 1$. Otherwise, the function result is $\varepsilon^{v_i}(\boldsymbol{I}_j) = 0$. Intuitively, the function resulting in $\varepsilon^{v_j}(\boldsymbol{I}_i)$ tells whether the data set $\mathcal{S}_i$ contains new data that has never been contained in the data set of the message that sensor v_j has transmitted.

To clarify the fourth bullet, a tree is given in Figure 6.8 to list all possible cases that a neighbor sensor of a non-application-member sensor can be.

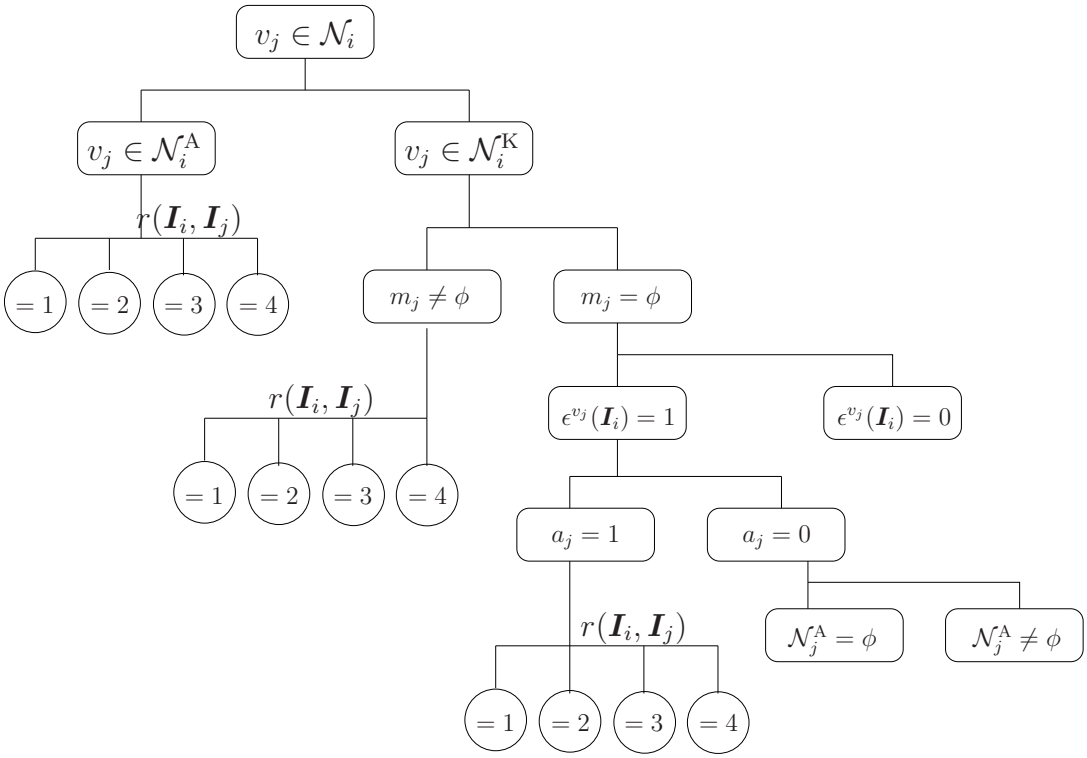

Figure 6.8. A condition tree to list of all possible cases based on the categorization conditions

Based on this tree in Figure 6.8, we can categorize the four types of feedback. Only if a neighbor sensor meets any of the situations for generating feedbacks, the neighbor sensor will expect I-Header or message transmissions from the center sensor from the center. The four feedback types T1, T2, T3, and T4 are given by tracing from the top of the tree to bottom by using colon ":" for each step down:

T1. $v_j \in \mathcal{N}_i$: $v_j \in \mathcal{N}_i^{\mathrm{A}}$: $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 2$ OR $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 4$;

T2. $v_j \in \mathcal{N}_i$: $v_j \in \mathcal{N}_i^{\mathrm{K}}$: $m_j \neq \phi$: $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 2$;

T3. $v_j \in \mathcal{N}_i$: $v_j \in \mathcal{N}_i^{\mathrm{K}}$: $m_j = \phi$: $\epsilon^{v_j}(\boldsymbol{I}_i) = 1$: $a_j = 1$:$r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 2$ OR $r(\boldsymbol{I}_i, \boldsymbol{I}_j) = 4$;

T4. $v_j \in \mathcal{N}_i$: $v_j \in \mathcal{N}_i^{\mathrm{K}}$: $m_j = \phi$: $\epsilon^{v_j}(\boldsymbol{I}_i) = 1$: $a_j = 0$: $\mathcal{N}_j^{\mathrm{A}} = \phi$;

For all other cases, no feedback is generated.

If sensor v_j gives T1 feedback, it is an application-member sensor. When the incoming I-Header indicates the existence of new data at the center sensor, the T1 feedback is sent, showing its interest in the incoming application message.

T2 feedback indicates that the non-application-member sensor v_j has an "older" message whose data set is a subset of the data set of the application message indicated by the I-Header transmitted by the center sensor.

When a neighbor sensor v_j sends T3 feedback, it is an agency sensor. The agency sensor compares the I-Header of the center with the stored I-Header in its memory and results in a positive result $\epsilon^{v_j}(\boldsymbol{I}_i) = 1$ indicating that there is new data which is not contained in any message that the neighbor has transmitted. Then it compares the received I-Header with the I-Header of the application-member sensor that it as an agency represents. If the comparison detects new data, the interest in the message will be given as T3 feedback from the agency sensor to the center.

T4 feedback is, on the other hand, sent by a non-agency sensor without application message. The sensor itself has no application-member neighbor sensors. When a non-application-member sensor having application-member neighbor sensors has no application message, and itself is not an agency sensor, it indicates that the application-member neighbor sensors of this non-application-member sensor have not finished the aggregation. It could be possible that after the aggregation being finished in its application-member neighbor sensors, the I-Header of the message indicates that it contains no less data than the I-Header received by the non-application-member sensor. To reduce the number of communications carried by the non-application-member sensors, this situation should be overcome by letting the discussed non-application-member sensor stay silence until the aggregation is finished among its application-member neighbor sensors.

6.3.5 Protocol for scenario 4

As discussed above, in Scenario 4, the center sensor helps its neighbor application-member sensors to exchange messages if necessary. Application-member sensors send

a *request* to v_i when they are waking up as the center. When v_i has received from every of its application-member neighbor sensor a request, i.e., $z_i = N_i^{\mathrm{A}}$, it initiates the communications. The protocol is given in Algorithm 14. Additionally, Algorithm 15 shows how sensor v_i find a neighbor sensor v_j, which contains the newest data.

Algorithm 14 Protocol for Scenario 4

if $z_i = N_i^{\mathrm{A}}$ **then**
  v_i initiates communication with its application-member neighbor sensors $\mathcal{N}_i^{\mathrm{A}}$
  For all sensor $v_j \in \mathcal{N}_i^{\mathrm{A}}$, v_j sends its I-Header $\boldsymbol{I}_j$ to sensor v_i
  if $\exists v_j, v_k \in N_i^{\mathrm{A}}, v_j \neq v_k$ such that $r(\boldsymbol{I}_j, \boldsymbol{I}_k) \neq 1$ **then**
    v_i applies algorithm 2 in Chapter 3 to find $\mathcal{P}^1$
    v_i finds a v_j according to the Algorithm 15
    $\boldsymbol{I}_i = \boldsymbol{I}_j$
    v_i informs v_j to send its message m_j
    v_j sends its message to v_i, $m_i = m_j$
    v_i broadcast I-Header $\boldsymbol{I}_i$ to all sensors in $\mathcal{N}_i^{\mathrm{A}}$
    v_i broadcast application message m_i to $\forall v_j \in \mathcal{N}_i^{\mathrm{A}}$
    $\forall v_j \in \mathcal{N}_i^{\mathrm{A}}$, v_j aggregates message m_i, performs bias-cancellation and updates its own message m_j and its own I-Header $\boldsymbol{I}_j$
    v_i updates its memory which stores the I-Header by $\Psi^{v_i} = \Psi^{v_i} \cup \boldsymbol{I}_j$, v_i clear its message $m_i = \phi$, its I-Header $\boldsymbol{I}_i = \mathbf{0}$ and resets the counter of received request $z_i = 0$
  else
    v_i resets the counter of received request $z_i = 0$
  end if
  v_i terminates communications with sensors in $\mathcal{N}_i^{\mathrm{K}}$
end if

Algorithm 15 Algorithm to find v_j which contains most new data

for $\mathcal{S}_l^{\mathrm{i}} \in \mathcal{P}^1$ **do**
  $k_l = 0$
  for $\mathcal{S}_m^{\mathrm{i}} \in \mathcal{P}^1 \setminus \mathcal{S}_l^{\mathrm{i}}$ **do**
    $\mathcal{S}_{l-m}^{\mathrm{i}} = \mathcal{S}_l^{\mathrm{i}} \cup \mathcal{S}_m^{\mathrm{i}} \setminus \mathcal{S}_m^{\mathrm{i}}$
    k_{l-m} is the number element in $\mathcal{S}_{l-m}^{\mathrm{i}}$
    if $k_{l-m} > k_l = 0$ **then**
      $k_l = k_{l-m}$
    end if
  end for
end for
Find v_j corresponding to the $\mathcal{S}^{\mathrm{i}} \in \mathcal{P}^1$ which results in the maximum k_l
Return v_j

6.3.6 Protocol for scenario 5

In Scenario 5, the center sensor is a non-application-member sensor with mixed application-member and non-application-member neighbor sensors. As discussed in the previous section, there are two tasks for the center sensor. The first is similar to the center sensor in Scenario 4 that the center helps its neighbor application-member sensors to exchange messages. The second is when all the application-member neighbor sensors have all aggregated the same data in their messages, the center sensor helps to forward this message to its non-application-member neighbor sensor(s) such that other application-member sensors can also receive this message.

There are two situations when a center sensor v_i in Scenario 5 wake up to initiate communications with its neighbor sensors. The first situation is that v_i has an application message which it received from another non-application-member sensor aiming to forward the message to other application-member sensors such as the application-member neighbor sensors of v_i. The second situation is when v_i receives enough number of Request from its application-member neighbor sensors. In this situation, v_i has no message. The protocol of Scenario 5 is given in Algorithm 16.

As seen in the protocol for Scenario 5, when the non-application-member center has messages, it interacts with all neighbor sensors that could potentially benefit from receiving the message of the center. When there is no message, it firstly acts as a sensor in Scenario 4. If all its neighbor application-member sensors have the same message, it becomes an agency sensor. As an agency sensor, it firstly interacts with other non-application-member neighbor sensors which can directly benefit from the message, i.e., the sensor with a message or another agency sensor. If there are no such neighbor sensors, it chooses a non-application-member sensor without messages and forwards the message from the application-member sensor to it. By doing this, the non-application-member sensors have less possibility to receive a message if they have currently no application message or they are not agencies.

6.3.7 Protocol for scenario 6

In Scenario 6, all sensors, including the center and its neighbor sensors, are non-application-member sensors. We may refer this scenario to the cases where messages are exchanged among application-member sensors which are isolated by intermediate non-application-member sensors. According to the rules we set in the previous section, the non-application-member sensor does not send a message if it does not wake up as

a center sensor. Therefore, one situation that the center sensor initiates the communications is that it has an application message. In the previous section, we have already pointed out a problem of "deadlock", that all sensors, including the center and all the neighbor sensors, have messages. We proposed a method that the center sensor may buffer its current message and behaves as a sensor ready for forwarding other application messages. Therefore, the second case that the center initiates the communications will be that the center has no message, but it has a message stored in the buffer. The algorithm for Scenario 6 is given in Algorithm 17.

6.4 Performance and discussions

In this section, we provide performance result through simulation and comparison against the benchmark scenario of our proposed protocols.

6.4.1 Benchmark

In order to test the proposed protocols given in the previous section, a benchmark for comparison is needed. The partial sensor involvement focuses on that only a part of sensors in the network are involved in the application, i.e., application-member sensors, to generate measurement and have interest in the function output. The sensors which are not involved in the application only help the message exchanges among the application-member sensors. The objectives of designing our protocol are to minimize the number of involved non-application-member sensors and to minimize the number of communications that are performed by those involved non-application-member sensors.

We design a benchmark scenario for comparison considers the following aspects:

- there are application-member sensors and non-application-member sensors in the wireless sensor network,
- the differences between application-member sensors and non-application-member sensors are that 1) the application-member sensors generate measurement data and the non-application-member sensors do not generate data, 2) the application-member sensors interest in the computation output,

- non-application-member sensors perform the computation and communication in the same way as the application-member sensor except that they do not initiate the communication when they have no application messages,
- The stop criterion is that all the application-member sensors have aggregated the measurement from all the application-member sensors in the network.

The benchmark considers the same application situations without utilizing the protocols we have been discussed in the previous section. Therefore, the protocols can be tested by comparing the two metrics, the number involved non-application-member sensor and the number of communications performed by the involved non-application-member sensors.

6.4.2 Performance

In the simulations, we consider a WSN with $N = 100$ sensors randomly deployed in a 1000-by-1000 squared area. The abbreviation RRG indicates Refined Random Gossiping proposed in this chapter. The performance is compared to the random gossiping (RG) proposed in Chapter 5. In the RG, the same sensors as the application-member sensors in the RRG generate measurement data, and the others generate no data. In contrast to RRG, all non-application-member sensors in the RG behave like application-member sensors, i.e., using the protocol defined in Scenario 1. The objectives of RG are 1) to compute the same divisible function as RRG and 2) to let the application-member sensors know the function output. For both approaches, the communications stop when all application-member sensors have the aggregated output function.

In Figure 6.9, the comparison is given by showing the number of communications in total T, the number of communications performed by application-member sensors T_{A} and the number of communications performed by non-application-member sensors T_{B} with respect to the communication range d. The ratio η_{A} of application-member sensors in the network is 0.2

In the Figure 6.9 d varies from the minimum value (approximately 200) that leads to a connected network to the value with which all sensors are connected to all the other sensors (the diagonal distance of the squared area). With almost the same total number T, the number T_{B} of communications for the non-application-member sensors using RRG is significantly reduced in comparison to that using RG. However,

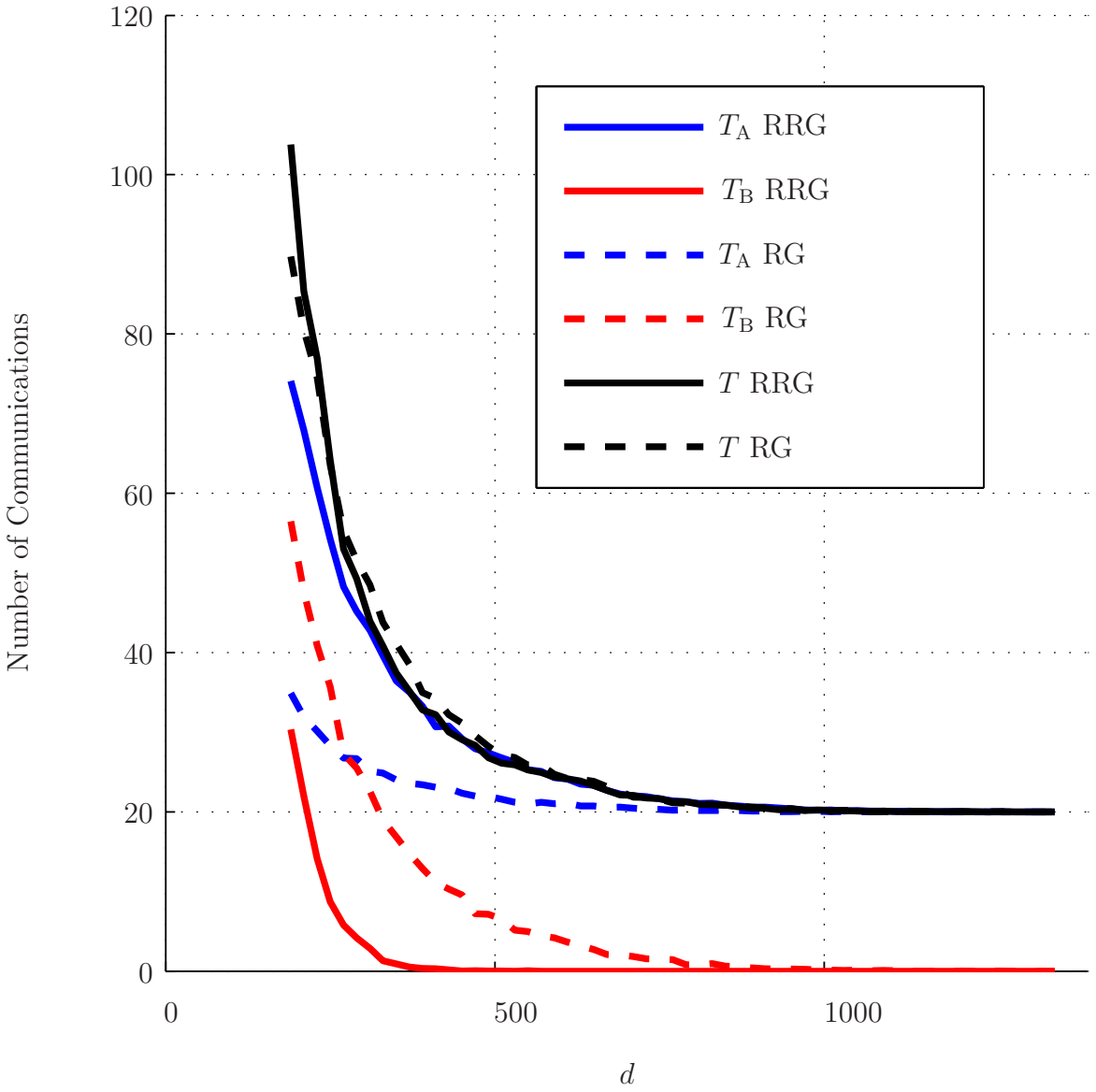

Figure 6.9. Number of communications vs. communication range d.

the price paid for such improvement is a higher number T_A of communications by application-member sensors. When increasing the communication range, a sensor has more neighbor sensors. Therefore, the probability that application-member sensors can directly communicate increases, which results in a decreased number of communications for both application-member sensors and non-application-member sensors.

Figure 6.10 compares for RG and RRG by showing the number of communications for different ratios η_A of the number of application-member sensors over the total number

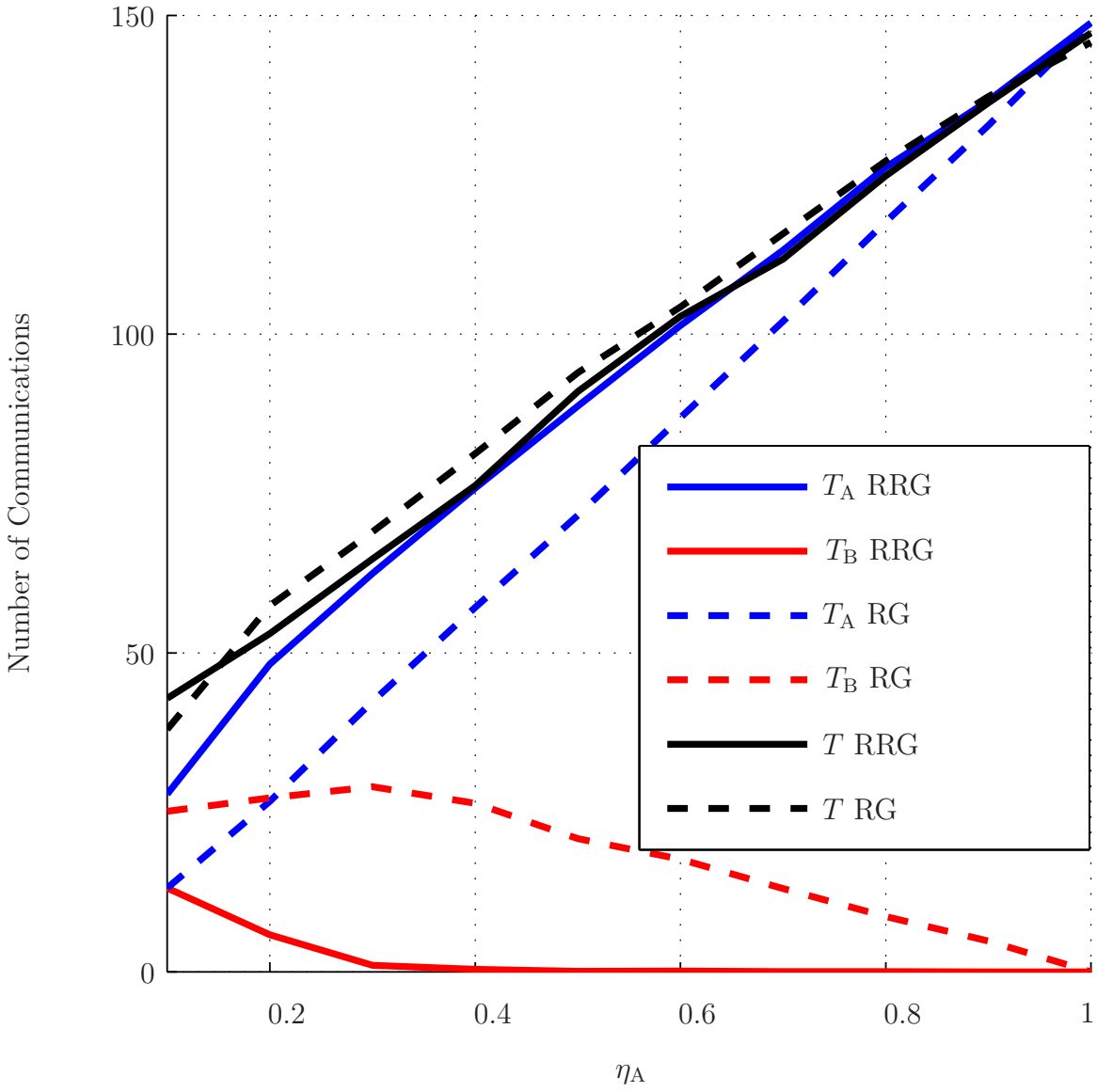

Figure 6.10. Number of communications vs. η_A

of sensors with a fixed communication range $d = 270$. As it is shown in the Figure, RRG outperforms RG in terms of the number of communications T_B performed by non-application-member sensors.

Figure 6.11 compares RG and RRG by the number of involved non-application-member sensors N_B in the message communications with respect to different communication ranges d. Figure 6.12 gives the comparison by the number of involved non-application-member sensors N_B with respect to the ratio η_A. In both Figures, the reduction of

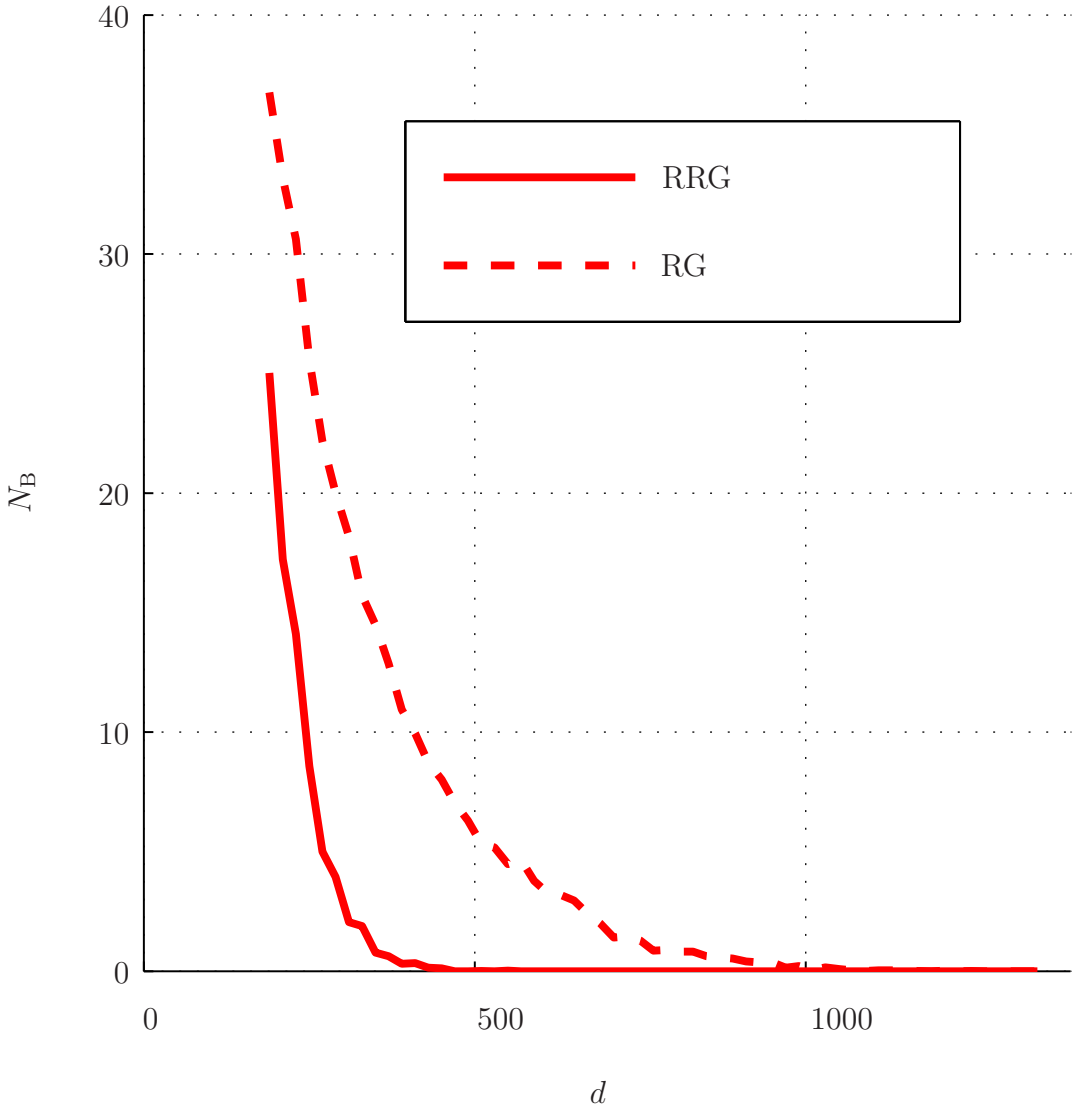

Figure 6.11. Number of involved non-application-member sensors N_B vs. communication range d

the involvement of non-application-member sensors are observed by using the RRG proposed in this chapter in comparison to the referred random gossiping protocols.

6.5 Summary

In this chapter, we propose a refined random gossiping protocol for wireless sensor networks where only part of the sensors are involved in the application, i.e., only a subset

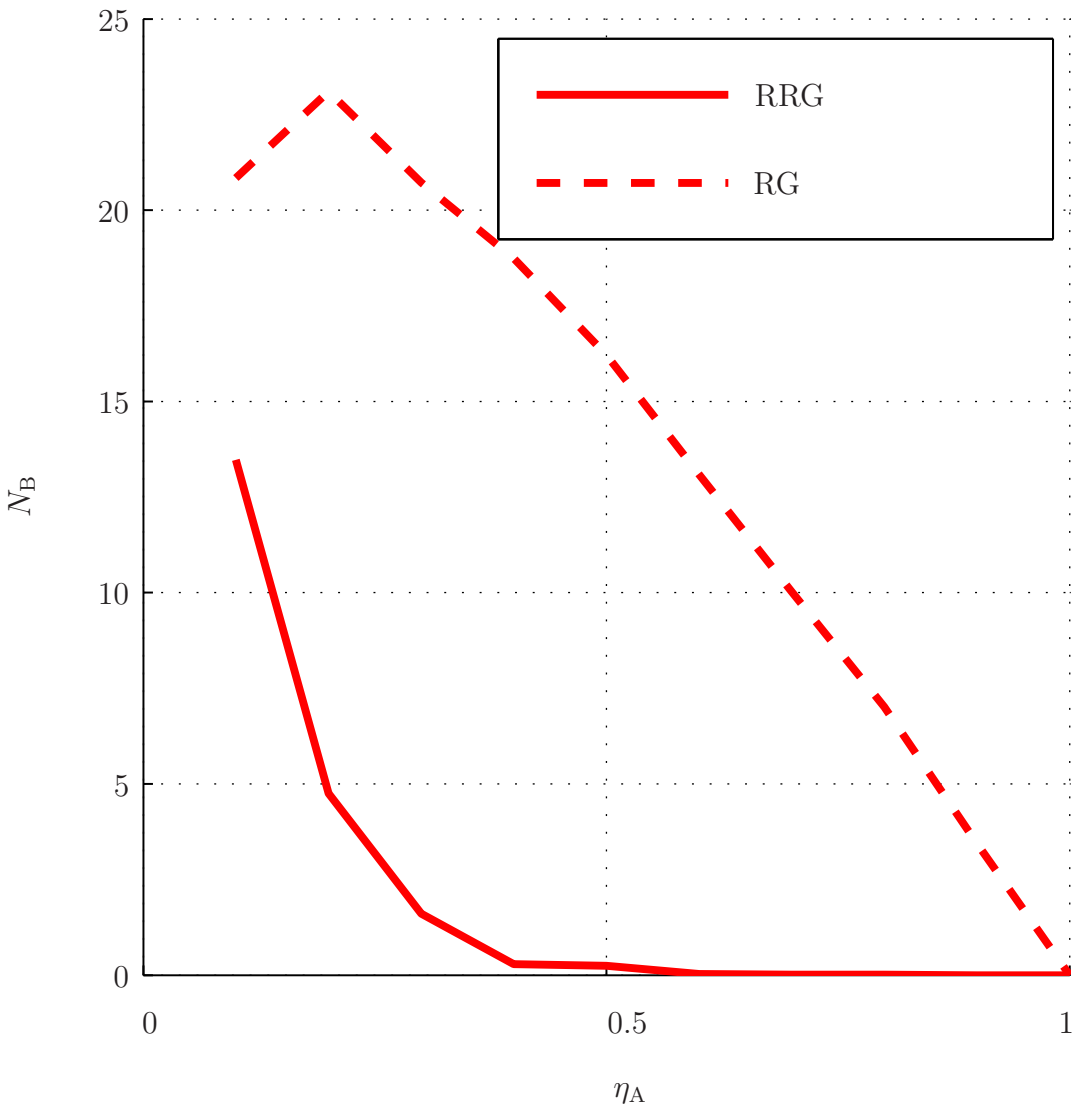

Figure 6.12. Number of involved non-application-member sensors N_B vs. Ratio η_A

of the sensors generates measurement data and is interested in the function output with the measurement data as parameters. Sensors are categorized into application-member sensors and non-application member sensors depending on their involvement in the application. Non-application-member sensors need to assist in the communications between application-member sensors. Our newly proposed refined protocol minimizes the number of involved non-application-member sensors and the number of their communications. Depending on the type of neighbor sensors, communication protocols for six different scenarios are discussed. Performance evaluations show the reduction of the number of communications performed by the non-application-member sensors as well as the number of non-application-member sensors that are involved in

the proposed protocols in comparison to the approach where all sensors communicate with each other as application-member sensors considered in our previous work.

Algorithm 16 Protocol for Scenario 5

1: **if** $m_i \neq \phi$ **then**
2: v_i initiates the communications with its neighbor sensors in $\mathcal{N}_i$
3: v_i broadcast its I-Header $\boldsymbol{I}_i$ to all its neighbor sensors $\mathcal{N}_i$
4: Sensors in $\mathcal{N}_i$ sends feedback to v_i if a feedback information is generated
5: **if** v_i receives any T1, T2 or T3 feedback **then**
6: v_i broadcast its message m_i to sensors $v_j \in \mathcal{N}_i$ which sent T1, T2 or T3 feedback
7: For v_j which sent T1 feedback, it aggregates the received message, performs bias-cancellation and updates its I-Header
8: For v_j which sent T2 feedback, it replaces its message with the received message and replaces its I-Header with the I-Header of its received message
9: For v_j which sent T3 feedback, it replaces its I-Header with the I-Header of its received message and set $\alpha_j = 0$
10: **end if**
11: v_i updates its memory which stores the I-Header by $\Psi^{v_i} = \Psi^{v_i} \cup \boldsymbol{I}_j$, v_i clears its message $m_i = \phi$, its I-Header $\boldsymbol{I}_i = \boldsymbol{0}$ and resets the counter of request $z_i = 0$
12: **else**
13: **if** $z_i = N_i^{\mathrm{A}}$ **then**
14: v_i initiates the communications with its neighbor sensors in $\mathcal{N}_i^{\mathcal{A}}$
15: For all sensor $v_j \in \mathcal{N}_i^{\mathrm{A}}$, v_j sends its I-Header $\boldsymbol{I}_j$ to sensor v_i
16: **if** $\exists v_j, v_k \in N_i^{\mathrm{A}}, v_j \neq v_k$ such that $r(\boldsymbol{I}_j, \boldsymbol{I}_k) \neq 1$ **then**
17: If v_i is no more an agency sensor if it was, $\alpha_i = 0$
18: Execute steps 5-13 in Algorithm 14
19: **else**
20: v_i is set to be an agency $\alpha_i = 1$, its I-Header is $\boldsymbol{I}_i = \boldsymbol{I}_j$ which is equal to any I-Header of its application-member neighbor sensor
21: v_i broadcast its I-Header $\boldsymbol{I}_i$ to all sensors in $\mathcal{N}_i^{\mathrm{K}}$
22: Sensor $v_j \in \mathcal{N}_i^{\mathrm{K}}$ send feedback to v_i if the feedback is generated
23: **if** v_i receives any T2 and T3 feedbacks **then**
24: v_i selects an $v_j \in \mathcal{N}_i^{\mathrm{A}}$ arbitrarily, v_j sends its message to v_i
25: v_i broadcast the message to a sensor in $v_j \in \mathcal{N}_i^{\mathrm{K}}$ which sent T2 and T3 feedback
26: For v_j which sent T2 feedback, it replaces its message with the received message and replaces its I-Header with the I-Header of its received message
27: For v_j which sent T3 feedback, it replaces its I-Header with the I-Header of its received message and set $\alpha_j = 0$
28: **else**
29: **if** v_i receives T4 feedbacks **then**
30: v_i selects an $v_j \in \mathcal{N}_i^{\mathrm{A}}$ arbitrarily, v_j sends its message to v_i
31: v_i randomly chooses one v_j which sent T4 feedback
32: v_i sends its message to v_j, v_j updates its message and its I-Header $\boldsymbol{I}_j = \boldsymbol{I}_i$
33: **end if**
34: **end if**
35: v_i updates its memory which stores the I-Header by $\Psi^{v_i} = \Psi^{v_i} \cup \boldsymbol{I}_j$, v_i clear its message $m_i = \phi$ and resets the counter of received request $z_i = 0$
36: **end if**
37: **end if**
38: **end if**

Algorithm 17 Protocol for Scenario 6

if $m_i = \phi$ **and** There is a message m_i^{S} in the buffer **then**
  v_i loads the message m_i^{S} from its buffer and set $m_i = m_i^{\mathrm{S}}$
  v_i sets its I-Header $\boldsymbol{I}_i$ as the I-Header of message m_i
  v_i clears its buffer for storing the message $m_i^{\mathrm{S}} = \phi$
end if
if $m_i \neq \phi$ **or** There is a message m_i^{S} in the buffer **then**
  v_i initiates the communications with its neighbor sensors in $\mathcal{N}_i$
  v_i broadcasts its I-Header $\boldsymbol{I}_i$ to all its neighbor sensors $\mathcal{N}_i$
  Sensors in $\mathcal{N}_i$ sends feedback to v_i if feedback information is generated
  if v_i receives no feedback from $\mathcal{N}_i$ **then**
    if $m_i^{\mathrm{S}} = \phi$ **then**
      v_i buffers its message such that $m_i^{\mathrm{S}} = m_i$ and clear its message $m_i = \phi$
    end if
  else
    if v_i receives any T2 or T3 feedback from $\mathcal{N}_i$ **then**
      v_i broadcasts the message to a sensor in $v_j \in \mathcal{N}_i$ which sent T2 or T3 feedback
      For v_j which sent T2 feedback, it replaces its message with the received message and replaces its I-Header with the I-Header of its received message
      For v_j which sent T3 feedback, it replaces its I-Header with the I-Header of its received message and set $\alpha_j = 0$
      v_i updates its memory which stores the I-Header by $\Psi^{v_i} = \Psi^{v_i} \cup \boldsymbol{I}_j$, v_i clear its message $m_i = \phi$
    else
      if v_i receives any T4 feedback from $\mathcal{N}_i$ **then**
        v_i randomly chooses one v_j which sent T4 feedback
        v_i sends its message to v_j, v_j updates its message and its I-Header $\boldsymbol{I}_j = \boldsymbol{I}_i$
        v_i updates its memory which stores the I-Header by $\Psi^{v_i} = \Psi^{v_i} \cup \boldsymbol{I}_j$, v_i clear its message $m_i = \phi$
      end if
    end if
  end if
end if

Chapter 7

Conclusions

In this chapter, a summary is given for this thesis.

This thesis presents our works on random gossiping in wireless sensor networks for divisible function calculations. Random gossiping is a decentralized communication protocol with which all sensors in the network can aggregate the measurements of all the other sensors.

Previous works on random gossiping focus on computing a weighted summation of the measurements of all sensors. The first problem in this thesis is to extend the random gossiping to compute general divisible functions not limited to the weighted summation. For a decentralized protocol, to ensure the convergence of the computation as well as to improve the speed of the convergence become critical problems. This is because there is no centralized scheduling in the network to create a sequential communication order for the data from a sensor to reach all other sensors within a given number of transmissions. The lack of centralized control in the network brings the additional problem of a biased aggregation, i.e., the measurement of a sensor may be aggregated multiple times by other sensors. When the relative position of a sensor regarding its neighbor sensors does not change in the network, the network has a static topology. A problem can be raised on random gossiping to make use of this topology condition to improve its convergence speed. Moreover, when multiple applications are running in a wireless sensor network, how to refine the random gossiping to support the multiple applications becomes a problem.

In this thesis, the foundation to solve these problems is the indicating header (I-Header) introduced in Chapter 2. Each message communicated by a sensor in the network is paired with an I-Header. An I-Header provides the information of a message on whether the message has aggregated the measurement of a particular sensor in the network. Therefore, a sensor can determine whether its message contains measurements of all sensors in the network, i.e., the convergence of the data aggregation is achieved, by using the I-Header paired with the message.

The I-Header itself contains no information regarding the aggregation bias. In Chapter 3, algorithms are proposed to reduce the aggregation bias at each sensor. The algorithms use the storage of sensors to keep some messages as well as the I-Headers of

the messages that have been previously communicated by a sensor and its neighbor sensors. When a sensor communicates with more than one neighbor sensor, the neighbor sensor selection is also considered as a part of the bias reduction algorithm. The I-Header provides the required information for sensors to perform the bias reduction algorithms.

Exchanging I-Headers among sensors can improve the convergence speed. In Chapter 4, protocols are given to describe how to integrate the I-Headers into random gossiping communications. The protocols are addressed based on the categorization of humble sensors and greedy sensors that are distinguished by how a sensor communicates with its neighbors. A humble sensor always exchanges its message with only one of its neighbor sensors at a time, whereas a greedy sensor exchanges its message with all of its neighbor sensors. A significant reduction in the number of message communications can be observed using the random gossiping protocols with I-Headers compared to the case without I-Headers. Furthermore, in Chapter 4, discussions are provided about the multihop coordination of random gossiping. The multihop coordination uses I-Headers to extend the sensor-neighbor communications to sensor-multihop-neighbor communications with a particular failure rate of the coordination. According to the performance analysis, multihop coordination of random gossiping improves the convergence speed significantly.

In Chapter 5, wireless sensor networks are considered with static topology when applying random gossiping. When the topology of the network is static, the neighbor sensors of a sensor do not change. Under this condition, algorithms are proposed to reduce the number of I-Header communications. Furthermore, the analysis shows that sensors at the topology bottleneck position of the network with static topology can apply transmission deferment to reduce the number of message communications in the network further.

In Chapter 6, the random gossiping is extended to support multiple applications in a wireless sensor network. The underlying assumption is that only a part of the total sensors in a network generates the measurements for an application and are interested in the aggregation results. Protocol refinement is proposed based on six different scenarios which are distinguished by whether a sensor and its neighbor sensors are involved in the application or not. The performance analysis shows that while maintaining the total number of communications, the refined protocol reduces the number of communications performed by sensors that are not involved in a specific application.

This thesis shows the benefit of introducing I-Headers and the gain in performance in terms of convergence speed and bias in the network. The main reason for the

improvement is the joint consideration of the network layer and the application layer. There are potential future works which can consider lower layers involving the network channel capacity analysis such as our works in [CKK+12a] and [CKK+12b]. Other potential future works may consider the coding of the message as well as the coding of the I-Header. The coding involves the entropy analysis of the I-Header itself. In a situation where the size of an I-Header is similar to that of a message, if the redundancy contained in the I-Headers can be reduced, the communication effort in the network for exchanging I-Headers will be reduced.

The area of wireless sensor networks is an enriching area for researches. Random gossiping and the use of I-Headers proposed in this thesis can also potentially be applied together with other cross-layer optimization approaches and computation offloading.

List of Acronyms

I-Header	Indicating-Header
IoT	Internet-of-Things
QoS	Quality-of-Service
RG	Random Gossiping
RRG	Refined Random Gossiping
SNR	Signal-to-Noise Ratio
WSN	Wireless Sensor Network

List of Symbols

$\boldsymbol{A}$	square matrix containing the connectivity among sensors
a_{ij}	entry on the i-th row and the j-th column of $\boldsymbol{A}$
$b_{m_i'}(s_m)$	bias in m_i' of measurement data s_m
$\mathcal{C}$	set collects all possible selections when select a group in $\mathcal{P}$
$\mathcal{C}_m$	m-th selection in $\mathcal{C}$
$\mathcal{C}_m(l)$	l-th data set in $\mathcal{C}_m$
$c_{\mathcal{S}_i}$	multiplicity of measurement data in multiset
$c_{\mathcal{S}_i}(j)$	multiplicity of data s_j being aggregated
D	maximum value of a finite square area
$\boldsymbol{D}$	diagonal matrix whose i-th diagonal entry equals N
d	communication range of every sensor in the network
d_{ij}	distance between the i-th sensor and the j-th sensor in a wireless sensor network
$\mathcal{E}$	set contains all connections between sensors in a wireless sensor network
e_{ij}	connection between sensor v_i and sensor v_j
$\mathcal{F}$	set of divisible function
f_l	function in set $\mathcal{F}$ with l input parameters
$\mathcal{G}$	wireless sensor network expressed using a graph
$g^{\Pi(\mathcal{S})}$	auxiliary function of the divisible function with partition $\Pi(\mathcal{S})$
$\boldsymbol{I}_i$	Indicating-Header of sensor v_i
$\boldsymbol{I}_i(j)$	j-th bit in $\boldsymbol{I}_i$
$\boldsymbol{L}$	Laplacian matrix of a network
$L_{\mathcal{S}^\Psi}$	maximum bias of the data in in $\mathcal{S}^\Psi$
l_k	cardinality of set $\mathcal{S}_k$
$m_l^{v_i}$	l-th message stored in sensor v_i
m_i'	new message at sensor v_i after aggregation
N	total number of sensors in a wireless sensor network
N_A	number of application member sensors
N_B	number of sensors in $\mathcal{V}_\mathrm{B}$
$\mathcal{N}_i$	set of neighbor sensors of sensor v_i
N_i	number of sensors in $\mathcal{N}_i$
N_K	number of non-application member sensors
$\mathbb{N}_{\geq 1}$	set of non-zero integer value

N^{av}	average number of neighbor sensors of each sensor in network
$\mathcal{N}_i^{\mathrm{S}}$	subset of the neighbor sensors of sensor v_i that intends to transmit data to v_i
$\mathcal{N}_i^{\Psi}$	set of sensors whose data set are in Ψ
$n_{\mathcal{C}}$	number of possible selections in $\mathcal{C}$
$n_{\mathcal{P}_j}$	number of sets of measurement data in $\mathcal{P}_j$
$\mathcal{P}$	set of groups generated by grouping sets of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$
$\mathcal{P}_j$	j-th group in $\mathcal{P}$
$\mathcal{P}_j(l)$	l-th set of measurement data in $\mathcal{P}_j$
$\mathcal{P}^1$	set collecting the first sets of measurement data of all groups in $\mathcal{P}$
p	number of groups in $\mathcal{P}$
r^{Θ}	function output index of the relation for comparing two data sets
$\mathcal{S}$	set of measurement data of all sensors
$\mathcal{S}_l$	l-th set of sensors' measurements after partition
$\mathcal{S}_{ij}^{\mathrm{B}}$	intersection of sets $\mathcal{S}_i$ and $\mathcal{S}_j$
$\mathcal{S}_{ij}^{\mathrm{iB}}$	set contains bias with operation of $\mathcal{S}_i^{\mathrm{i}} \cap \mathcal{S}_j^{\mathrm{i}}$
$\mathcal{S}_i^{\mathrm{i}}$	underlying set of measurement data in multiset
$\mathcal{S}_i^{\mathrm{i}'}$	set of measurement data in accumulation of measurement data of sensor in $\mathcal{N}_i^{\mathrm{S}}$ and measurement data of sensor v_i
$(\mathcal{S}_i^{\mathrm{i}}, c_{\mathcal{S}_i})$	multiset for data aggregation at sensor v_i
$\mathcal{S}_i^{\mathrm{R}}$	reference data set of aggregation at sensor v_i
$\mathcal{S}^{\mathcal{T}}$	data set selected to perform bias cancellation
$\mathcal{S}_{ij}^{\mathrm{UB}}$	unbiased version of $\mathcal{S}_{ij}^{\mathrm{B}}$
$\mathcal{S}_l^{v_i}$	l-th set measurement data of which the aggregation is available at sensor v_i
$\mathcal{S}_{1\to i}^{v_i}$	set accumulates from $\mathcal{S}_1^{v_i}$ to $\mathcal{S}_i^{v_i}$
$\boldsymbol{s}_{\mathcal{S}_l^{v_i}}$	data vector of set of measurement data of $\mathcal{S}_l^{v_i}$
$\boldsymbol{s}$	vector collects all measurement data in $\mathcal{S}$
$\boldsymbol{s}_{\mathcal{S}_{1\to i}^{v_i}}$	data vector corresponding to $\mathcal{S}_{1\to i}^{v_i}$
$\bar{s}$	mean value of all sensors' measurement data
$\boldsymbol{s}_{\mathcal{S}_k}$	vector collects all measurement data in set $\mathcal{S}_k$
s_i	measurement data of sensor v_i
$\mathcal{T}$	combination table
T	total number of message communications performed in network
T_{A}	number of message communications performed by sensors in $\mathcal{V}_{\mathrm{A}}$
T_{B}	number of message communications performed by sensors in $\mathcal{V}_{\mathrm{B}}$

$\mathcal{V}$	set contains all sensors in a wireless sensor network
$\mathcal{V}_{\mathrm{A}}$	set of application member sensors
$\mathcal{V}_{\mathrm{B}}$	set of non-application member sensors which assist communications
$\mathcal{V}_{\mathrm{K}}$	set of non-application member sensors
$\mathcal{V}_{s_i}^{l}$	a set of l sensors which have already aggregated data s_i
v	sensor in a wireless sensor network
v_i	i-th sensor in a wireless sensor network
v^{s}	selected sensor
x, y	coordinate position in a finite area
x_i	aggregation data of sensor v_i
Θ	map function from Indicating Header to sensor ID
Λ_{Ω}	set collects all possible permutations of data sets in Ω
λ_2	second smallest eigenvalue of $\boldsymbol{L}$
$\Pi(\mathcal{S})$	partition of $\mathcal{S}$
τ_i^l	lower boundary of a value range
τ_i^u	upper boundary of a value range
Υ	a multiset operation which is either union of set-theoretic difference
Υ_i	operation between set $\mathcal{S}_i^{v_i}$ and set $\mathcal{S}_{i+1}^{v_i}$
ϕ	empty set
Ψ	selection in $\mathcal{C}$
$\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$	ordered set collects all set of measurement data of the messages at sensors in $\mathcal{N}_i^{\mathrm{S}}$
$\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}(l)$	l-th set of measurement data in $\Psi_i^{\mathcal{N}_i^{\mathrm{S}}}$
Ψ^{v_i}	set collects ψ_i sets of measurement data which are available at v_i
ψ_i	number of messages stored in buffer of sensor v_i
Ω	set of data set in Ψ^{v_i}
$\Omega(i)$	i-th data data set in Ω
Ω_b	bias reduction set
Ω^x	permutation defined by one possibility in Λ_{Ω}
$\coprod$	collection of all possible choices of Υ

Bibliography

[AA09] A. Alomar and A. Akbar, „CЯC: Carrefour Routing for Cluster-based Wireless Sensor Network to Support Multiple Applications in Multiple Gateway Environments“, in *Proc. International Conference on Mobile Ad-hoc and Sensor Networks*, 2009.

[AIM10] L. Atzori, A. Iera, and G. Morabito, „The internet of things: A survey“, *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[AK04] J. Al-Karaki and A. Kamal, „Routing techniques in wireless sensor networks: a survey“, *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[AMC07] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, „Wireless multimedia sensor networks: A survey“, *IEEE Wireless Communication Magazine*, vol. 14, no. 6, pp. 32–39, 2007.

[Arn02] S. Arnon, „Collaborative network of wireless microsensors“, *Electronics Letters*, vol. 36, no. 2, pp. 186–187, 2002.

[Arn05] ——, „Deriving an upper bound on the average operation time of a wireless sensor network“, *Communications Letters*, vol. 9, no. 2, pp. 154–156, 2005.

[ASSC02] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, „A Survey on Sensor Networks“, *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[AYSS09] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, „Broadcast Gossip Algorithms for Consensus“, *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2748–2761, 2009.

[BGPS04] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, „Analysis and optimization of randomized gossip algorithms“, in *Proc. 43rd IEEE Conference on Decision and Control*, 2004.

[BGPS05] ——, „Gossip algorithms: Design, analysis, and applications“, in *Proc. 24th Annual IEEE International Conference on Computer Communications*, 2005.

[BGPS06] ——, „Randomized Gossip Algorithms“, *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[Böh12] T. M. Böhler, „Industrie 4.0 - Smarte Produkte und Fabriken revolutionieren die Industrie“, *Produktion Magazin*, 2012.

[BS03] A. Boulis and M. Srivastava, „Node-level Energy Management for Sensor Networks in the Presence of Multiple Applications“, in *Proc. International Conference on Pervasive Computing and Communications*, 2003.

[BSLR10] S. Bhattacharya, A. Saifullah, C. Lu, and G.-C. Roman, „Multi-Application Deployment in Shared Sensor Networks based on Quality of Monitoring“, in *Proc. Real-Time and Embedded Technology and Applications Symposium*, 2010.

[Chu97] F. R. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[CK03] C.-Y. Chong and S. Kumar, „Sensor networks: evolution, opportunities, and challenges“, *Proceedings of the IEEE*, vol. 91, no. 8, pp. 1247–1256, 2003.

[CKK+12a] Z. Chen, A. Kuehne, A. Klein, A. Loch, M. Hollick, and J. Widmer, „Two-Way Relaying for Multiple Applications in Wireless Sensor Networks“, in *Proc. International ITG Workshop on Smart Antennas*, 2012.

[CKK+12b] ——, „Two-Way Relaying for Multiple Applications in Wireless Sensor Networks“, in *Proc. International ITG Workshop on Smart Antennas*, 2012.

[CKK13a] Z. Chen, A. Kuehne, and A. Klein, „Multi-hop Coordination in Gossiping-based Wireless Sensor Networks“, in *Proc. 10th International Symposium on Wireless Communication Systems*, 2013.

[CKK13b] ——, „Reducing Aggregation Bias and Time in Gossiping-based Wireless Sensor Networks“, in *Proc. 14th IEEE International Workshop on Signal Processing Advances in Wireless Communciations*, 2013.

[CKK14a] ——, „Improved Bias Cancellation and Header Communication for Random Gossiping-based Static Wireless Sensor Networks“, in *Proc. 18th International ITG Workshop on Smart Antennas*, 2014.

[CKK14b] ——, „Randomized Gossip Protocol in Wireless Sensor Networks with Partial Sensor Involvement“, in *Proc. 11th International Symposium on Wireless Communication Systems*, 2014.

[CMH10] D. Christin, P. Mogre, and M. Hollick, „Survey on Wireless Sensor Network Technologies for Industrial Automation: The Security and Quality of Service Perspectives“, *Future Internet*, vol. 2, no. 2, pp. 96–125, 2010.

[CSA13] Y. Chen, S. Shakkottai, and J. Andrews, „On the Role of Mobility for Multimessage Gossip“, *Transactions on Information Theory*, vol. 59, no. 6, pp. 3953–3970, 2013.

[CW06] M. Cardei and J. Wu, „Energy-Efficient Coverage Problems in Wireless Ad-Hoc Sensor Networks“, *Computer Communications*, vol. 29, no. 4, pp. 413–420, 2006.

[DBS11] P. Di Lorenzo, S. Barbarossa, and A. Sayed, „Bio-inspired swarming for dynamic radio access based on diffusion adaptation“, in *Proc. 20th European Signal Processing Conference*, 2011.

[DSW08] A. Dimakis, A. Sarwate, and M. Wainwright, „Geographic Gossip: Efficient Averaging for Sensor Networks“, *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 1205–1216, 2008.

[EXR+11] N. Edalat, W. Xiao, N. Roy, S. Das, and M. Notani, „Combinatorial Auction-Based Task Allocation in Multi-Application Wireless Sensor Networks“, in *Proc. International Conference on Embedded and Ubiquitous Computing*, 2011.

[FKK10] N. Freris, H. Kowshik, and P. Kumar, „Fundamentals of Large Sensor Networks: Connectivity, Capacity, Clocks, and Computation“, *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1828–1846, 2010.

[FKS11] M. Farooq, T. Kunz, and M. St-Hilaire, „Cross Layer Architecture for Supporting Multiple Applications in Wireless Multimedia Sensor Networks“, in *Proc. International Wireless Communications and Mobile Computing Conference*, 2011.

[GBS12] M. Goldenbaum, H. Boche, and S. Stanczak, „Nomographic gossiping for f-consensus“, in *Proc. 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, 2012.

[GHSW11] P. Glatz, L. Hoermann, C. Steger, and R. Weiss, „MAMA: Multi-ApplicationMiddlewAre for Efficient Wireless Sensor Networks“, in *Proc. 18th International Conference on Telecommunications*, 2011.

[GK05] A. Giridhar and P. R. Kumar, „Computing and Communicating Functions Over Sensor Networks“, *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 755–764, 2005.

[GK06] ——, „Toward a theory of in-network computation in wireless sensor networks“, *IEEE Communications Magazine*, vol. 44, no. 4, pp. 98–107, 2006.

[HHL88] S. Hedetniemi, S. Hedetniemi, and A. Liestman, „A Survey of Gossiping and Broadcasting in Communication Networks“, *Networks*, vol. 18, pp. 319–349, 1988.

[JHI07] A. Jayasumana, Q. Han, and T. Illangasekare, „Virtual Sensor Networks - A Resource Efficient Approach for Concurrent Applications“, in *Proc. International Conference on Information Technology*, 2007.

[KDG03] D. Kempe, A. Dobra, and J. Gehrke, „Gossip-based computation of aggregate information“, in *Proc. 44th Annual IEEE Symposium on Foundations of Computer Science*, 2003.

[KK02] D. Kempe and J. Kleinberg, „Protocols and impossibility results for gossip-based communication mechanisms“, in *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002.

[KKD01] D. Kempe, J. Kleinberg, and A. Demers, „Spatial gossip and resource location protocols“, in *Proc. 33rd ACM symposium on Theory of Computing*, 2001.

[KLS08] S. Kraus, R. Lin, and Y. Shavitt, „On Self-Interested Agents in Vehicular Networks With Car-to-Car Gossiping“, *Transactions on Vehicular Technology*, vol. 57, no. 6, pp. 3319–3332, 2008.

[KLT03] M. Karpovsky, L. Levitin, and A. Trachtenberg, „Data verification and reconciliation with generalized error-control codes“, *Transactions on Information Theory*, vol. 49, no. 7, pp. 1788–1793, 2003.

[KMN11] N. Kapoor, S. Majumdar, and B. Nandy, „Scheduling on Wireless Sensor Networks Hosting Multiple Applications“, in *Proc. International Conference on Communications*, 2011.

[KMN12] ——, „System and Application Knowledge Based Scheduling of Multiple Applications in a WSN“, in *Proc. International Conference on Communications*, 2012.

[KSSV00] R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking, „Randomized rumor spreading“, in *Proc. 41st Annual Symposium on Foundations of Computer Science*, 2000.

[LCS13] Y. Lin, S. Chang, and H. Sun, „CDAMA: Concealed Data Aggregation Scheme for Multiple Applications in Wireless Sensor Networks“, *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 7, pp. 1471–1483, 2013.

[LDZ+08] W. Li, Y. Du, Y. Zhang, B. Childers, P. Zhou, and J. Yang, „Adaptive Buffer Management for Efficient Code Dissemination in Multi-Application Wireless Sensor Networks“, in *Proc. International conference on Embedded un Ubiquitous Computing*, 2008.

[LM09] A. Lamsa and J. Mantyjarvi, „Social networking service for mobile communities based on spatial cumulative gossiping“, in *3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*, 2009.

[MS08] D. Mosk-Aoyama and D. Shah, „Fast Distributed Algorithms for Computing Separable Functions“, *IEEE Transactions on Information Theory*, vol. 54, no. 7, pp. 2997–3007, 2008.

[MZ10] A. Majeed and T. Zia, „Multi-Set Architecture for Multi-Applications Running on Wireless Sensor Networks“, in *Proc. International Conference on Advanced Information Networking and Applications Workshops*, 2010.

[RRJ10] N. Roy, V. Rajamani, and C. Julien, „Supporting Multi-Fidelity-Aware Concurrent Applications in Dynamic Sensor Networks“, in *Proc. IEEE International Conference on Pervasive Computing and Communications Workshops*, 2010.

[RS11] S. Rajagopalan and D. Shah, „Distributed Averaging in Dynamic Networks“, *Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 845–854, 2011.

[RV06] R. Rajagopalan and P. K. Varshney, „Data-Aggregation Techniques in Sensor Networks: A Survey“, *IEEE Communications Survey and Tutorials*, vol. 8, no. 4, pp. 48–63, 2006.

[SBS12] S. Sardellitti, S. Barbarossa, and A. Swami, „Optimal Topology Control and Power Allocation for Minimum Energy Consumption in Consensus Networks“, *IEEE Transactions on Signal Processing*, vol. 60, no. 1, pp. 383–399, 2012.

[SEH11] M. Shazly, E. S. Elmallah, and J. Harms, „Balanced Slices in Wireless Sensor Networks“, in *Proc. Wireless Communications and Networking Conference*, 2011.

[SEH12] M. Shazly, E. Elmallah, and J. Harms, „Balancing Area Coverage in Partitioned Wireless Sensor Networks“, in *Proc. Wireless Communications and Networking Conference*, 2012.

[SH08] S. Sundaram and C. Hadjicostis, „Distributed function calculation and consensus using linear iterative strategies“, *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 650–660, 2008.

[SKP+11] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, „Smart Cities and the Future Internet: Towards Cooperation Frameworks for Open Innovation“, *The Future Internet, Lecture Notes in Computer Science*, vol. 6656, pp. 431–446, 2011.

[WVMX14] Y. Wang, A. Vasilakos, J. Ma, and N. Xiong, „On Studying the Impact of Uncertainty on Behavior Diffusion in Social Networks“, *Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 2, pp. 185–197, 2014.

[XHL14] L. Xu, W. He, and S. Li, „Internet of Things in Industries: A Survey“, *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, 2014.

[XSC+10] Y. Xu, A. Saifullah, Y. Chen, C. Lu, and S. Bhattacharya, „Near Optimal Multi-Application Allocation in Shared Sensor Networks“, in *Proc. 11th ACM international symposium on Mobile ad hoc networking and computing*, 2010.

[ZBC+14] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, „Internet of Things for Smart Cities“, *Internet of Things*, vol. 1, no. 1, pp. 22–32, 2014.

Own Publications

[CKK+12a] Z. Chen, A. Kuehne, A. Klein, A. Loch, M. Hollick, and J. Widmer, „Two-Way Relaying for Multiple Applications in Wireless Sensor Networks“, in *Proc. International ITG Workshop on Smart Antennas*, 2012.

[CKK+12b] ——, „Two-Way Relaying for Multiple Applications in Wireless Sensor Networks“, in *Proc. International ITG Workshop on Smart Antennas*, 2012.

[CKK13a] Z. Chen, A. Kuehne, and A. Klein, „Multi-hop Coordination in Gossiping-based Wireless Sensor Networks“, in *Proc. 10th International Symposium on Wireless Communication Systems*, 2013.

[CKK13b] ——, „Reducing Aggregation Bias and Time in Gossiping-based Wireless Sensor Networks“, in *Proc. 14th IEEE International Workshop on Signal Processing Advances in Wireless Communciations*, 2013.

[CKK14a] ——, „Improved Bias Cancellation and Header Communication for Random Gossiping-based Static Wireless Sensor Networks“, in *Proc. 18th International ITG Workshop on Smart Antennas*, 2014.

[CKK14b] ——, „Randomized Gossip Protocol in Wireless Sensor Networks with Partial Sensor Involvement“, in *Proc. 11th International Symposium on Wireless Communication Systems*, 2014.

Supervised Student Theses

Name	Title of the thesis	Thesis type	Date
Moawad, Michael	The combination of Random Gossiping and Clustering in Wireless Sensor Networks	Bachelor Thesis	07/2014
Armanious, Karim	Throughput-Lifetime Trade-Offs in Wireless Sensor Networks	Bachelor Thesis	07/2013
Gune, Abhijeet	Routing and Scheduling in WSN with Simultaneous Communications	Master Thesis	07/2013

Erklärungen laut Promotionsordnung

§8 Abs. 1 lit. c PromO
Ich versichere hiermit, dass die elektronische Version meiner Dissertation mit der schriftlichen Version übereinstimmt.

§8 Abs. 1 lit. d PromO
Ich versichere hiermit, dass zu einem vorherigen Zeitpunkt noch keine Promotion versucht wurde. In diesem Fall sind nähere Angaben über Zeitpunkt, Hochschule, Dissertationsthema und Ergebnis dieses Versuchs mitzuteilen.

§9 Abs. 1 PromO
Ich versichere hiermit, dass die vorliegende Dissertation selbstständig und nur unter Verwendung der angegebenen Quellen verfasst wurde.

§9 Abs. 2 PromO
Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

19.07.2019 Zhiliang Chen

Datum und Unterschrift